(NASA-CR-171138) REFLIGHT CERTIFICATION SOFTWARE DESIGN SPECIFICATIONS (Intermetrics, Inc.) 143 p HC A07/MF A01 CSCL 09B
N84-31966
Unclas
G3/61 21941

INTERMETRICS

IR-AL-032

PDSS/IMC
REFLIGHT CERTIFICATION SOFTWARE
DESIGN SPECIFICATIONS

15 AUGUST 1984

CONTRACT NUMBER: NAS8-33825

Prepared For: National Aeronautics and Space Administration
George C. Marshall Space Flight Center
Marshall Space Flight Center, Alabama 35812

Prepared By: Intermetrics, Inc.
3322 South Memorial Parkway
Huntsville, Alabama 35801
(205) 883-6860

RELEASE
DATE:

DOCUMENT CHANGE RECORD

SHEET:

CHANGE DOCUMENT:	IR-AL-032 PDSS/IMC Reflight Certification Software Design Specifications (*NUMBERED*)	*REVISION*	15 Aug 1984 (*DATED*)	*COPY NO.*

CHANGE (DATE)	TO INCORPORATE THE CHANGE INTO THE DOCUMENT: 1. ADD PAGES:	2. REMOVE PAGES:	OTHER COMMENTS REGARDING THIS CHANGE (AUTH., INSTRUCT., ETC.)
1-Jun-84	Review Copy (CDR)		
15-Aug-84	Revision 1.0 Release		IR-32-002, IR-32-003, IR-32-005, IR-32-007, IR-32-007, IR-32-009, IR-32-010, IR-32-011, IR-32-013

PREFACE

This document contains the PDSS/IMC Software Design Specification for the Payload Development Support System (PDSS)/Image Motion Compensator (IMC). The PDSS/IMC is to be used for checkout and verification of the IMC flight hardware and software by NASA/MSFC.

This document was prepared for the Information and Electronic Systems Laboratory of the Marshall Space Flight Center under NASA contract NAS8-33825.

Technical direction was provided by:

Mr. Jim Lewis (EB32)
Mr. Bob Panciera (EB32)
Mr. Ken Williamson (EB42)

Questions concerning this document should be directed to the Intermetrics, Inc., Huntsville office.

Intermetrics, Inc.
3322 South Memorial Parkway
Century Office Center, Suite 72
Huntsville, Alabama 35801
(205) 883-6860

Approved
By: J. R. Bounds
J. R. Bounds
Director
Southeast Division

TABLE OF CONTENTS

TABLE OF CONTENTS
(CONTINUED)

TABLE OF CONTENTS
(CONTINUED)

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

AI	Analog Input
AO	Analog Output
AST	Astros Start Tracker
ASTROS	Advanced Star/Target Reference Optical Sensor
CCD	Charged Coupled Device
CDMS	Command and Data Management System
CIS	Computer Interface Simulation
CPD	Cruciform Power Distributor
DEP	Dedicated Experiment Processor
DI	Discrete Input
DO	Discrete Output
DRIRU	Dry Rotor Inertial Reference Unit
ECAS	Experiment Computer Application Software
ECIO	Experiment Computer Input/Output
ECOS	Experiment Computer Operating System
ESA	European Space Agency
GML	General Measurement Loop
GMT	Greenwich Mean Time
GSE	Ground Support Equipment
HRM	High Rate Multiplexor
HUT	Hopkins Ultraviolet Telescope
IIA	Instrument Interface Agreement
IMC	Image Motion Compensator
IMCE	Image Motion Compensation Electronics
IMCS	Image Motion Compensation Subsystem
IPS	Instrument Pointing System
LAM	Look At Me
MMU	Mass Memory Unit
NASA	National Aeronautics and Space Administration
PCC	Programmable Crate Controller
PCM	Pulse Code Modulated
pid	Page Identifier
PDSS	Payload Development Support System
POCC	Payload Operations Control Center
QT	Qualification Test
RAU	Remote Acquisition Unit
RAUI	Remote Access Unit Interface
RAUS	Remote Access Unit Simulator
RFC	Reflight Certification
RIUI	Remote Interface Unit Interface
SEID	Spacelab Experiment Interface Device
SI	Serial Input
sid	Signal Identifier
SL	Space Lab
SO	Serial Output
SPL	Scratch Pad Line
SPSME	Spacelab Payload Standard Modular Electronics

ACRONYMS
(CONTINUED)

SRR	Software Requirements Review
SWCDR	Software Critical Design Review
SWCI	Software Configuration Inspection
SWPDR	Software Preliminary Design Review
UV	Ultraviolet
UIT	Ultraviolet Imaging Telescope
UTC	User Time Clock
WUPPE	Wisconsin Ultraviolet Photopolarometry Experiment

1.0 INTRODUCTION

The Reflight Certification software described herein has been designed in accordance with the "IMCS Reflight Certification Requirements and Design Specifications", IR-AL-016, 29 January 1984.

The Reflight Certification software has been designed as an application task on the Payload Development Support System (PDSS) which utilizes the Spacelab Experiment Interface Device (SEID) RAU simulator.

The Reflight Certification software has been designed to certify the IMCS interfaces IMCE/WUPPE, IMCE/AST, IMCE/UIT, IMCE/DRIRU, IMCE/HRM, and IMCE/RAU.

The remainder of section 1.0 describes the overall structure of the IMCS hardware and software. Section 2.0 contains the RFC task specifications. Section 3.0 contains the RFC routine specifications. Section 4.0 contains the RFC data specifications. Appendix A contains the IMCE figures. Appendix B contains data tables for IMCS Reflight Certification data. Appendix C contains PDSS/IMC Reflight Certification user's operating information.

1.1 SYSTEM STRUCTURE

Figure A-1 shows the IMCE Functional Layout with interfaces. Of importance for Reflight Certification are the interfaces between IMCE and the Spacelab CDMS that includes the

Remote Acquisition Unit, Time Interface Module, and High Rate Mux. Figure A-2 is a black box representation of the IMCS.

Figure A-3 will be the configuration of the PDSS/IMC for performing Reflight Certification. The PDSS/IMC will provide those functions normally provided by the Experiment Computer, ECOS, and IMCS ECAS.

The PDSS IMC system interfaces to the IMCE are the SEID RAUI serial channel for SPSME and DEP protocol and the SEID Discrete Outputs to the Cruciform Power Distribution unit.

1.2 SOFTWARE STRUCTURE

The IMCE RFC application software is invoked by the user via the INIT command from the PDSS master display. The application will load and start the IMCE SEID GML monitor. The PDSS software acquires the SEID GML data and places the data in the SEID data buffers. Once started, the IMCE RFC software retains control of the PDSS. Section 2.0 describes the RFC user tasks.

The principal source of data for the Reflight Certification software is the ECIO data acquired by the SEID from the IMCE using SPSME protocol. See Figure A-4 for a data flow diagram. The IMCE ECIO data stream is identical to the flight ECIO data stream.

1.3 IMCS DISPLAYS

The PDSS/IMC application provides the capability to define up to five display pages for the CONRAC. The main display will

be a simulation of the IMCS Flight Crew page (see Figure A-6). The PDSS master display is also available (see Figure A-7). The pages that have been defined are listed in the table below.

TABLE 1-1: DISPLAY PAGES

ID	NAME	FIGURE	DESCRIPTION
1	D.001	8	IMCS DDU Flight Page
2	D.002	9	Aux Flight Page
3	D.003	10	Special Data
4	D.004	11	IMC ECIO Data
5	D.005	12	Power

Section 4.0 contains a description of the display data bases that drive these displays.

1.4 PROGRAM DESCRIPTION LANGUAGE OVERVIEW

The detailed design of the PDSS/IMC Reflight Certification software is described in a natural high level program design language (pdl). The form of the pdl is similar to PASCAL or other structured programming languages. The pdl is simple to read, easy to understand, and is adaptable to any programming language. The amount of detail contained in the pdl is left to the user.

1.4.1 SPECIAL WORD MEANINGS

Some words have special meaning for the pdl.

bic - bit clear (LSI 11/23 operation)
bis - bit set (LSI 11/23 operation)
bit - bit test (LSI 11/23 operation)
rjs - right justified
ljs - left justified
pdl - program description language

1.4.2 DETAILED CONSTRUCTS

Program implementation details and comments are enclosed in "|- -|".

Example 1:

set log flag off |-XLOG=0-|

The log flag 'XLOG" is set to zero (the off state).

Example 2:

post start event |-POST(EVSTRT)-|

The start event "EVSTRT" is posted by calling the POST routine with EVSTRT as its input parameter.

Example 3:

clear F code in CSR |-bic (#7,CSR)-|

The F code in the CSR is cleared by performing a bit-clear operation on CSR with the octal data pattern #7.

1.4.3 BASIC CONSTRUCTS

The following basic constructs are used by the pdl.

(a)
```
IF condition
   then
      statement(s)
   else
      statement(s)
END-IF
```

(b)
```
DO UNTIL condition
      statement(s)
END-DO
```

(c)
```
DO FOR index
      statement(s)
END-DO
```

(d)
```
CASE index of object
   index-1:
      statement(s)
   index-2:
      statement(s)
      .
      .
      .
   index-n:
      statement(s)
   else:
      statement(s)
END-CASE
```

(e) ROUTINE=name

```
ROUTINE=name
      statement(s)
END-ROUTINE
```

(f) TASK=name

```
TASK=name
      statement(s)
END-TASK
```

2.0 RFC SOFTWARE DESCRIPTION

The Reflight Certification software is comprised of the PDSS software package, the SEID software package, and the PDSS/IMC Reflight Certification (RFC) software application package. The PDSS software package is described in the PDSS User's Manual (IR-AL-001, Revision 2.1, 15 July 1984) and the SEID software package is defined in the SEID II Specification (IR-AL-007, Revision 1.0, 15 July 1984).

The PDSS/IMC RFC software package is implemented as user tasks under PDSS. The following user tasks compose the PDSS/IMC RFC software package.

USR	NAME	RATE	FUNCTION
USR20	EXEC	1HZ	Executive
USR21	COMTRK	10HZ	Comet Track
USR22	CREW	1HZ	Process Crew Commands
USR23	FLTDIS	1HZ	Update DDU Page
USR24	EXMON	1HZ	Exception Monitor
USR25	ECAS	1HZ	Perform IMCS ECAS
USR26	TLOGER	1HZ	Log Function
USR27	QTDISP	1HZ	Update IMCE Displays
USR28	QTKB	1HZ	Keyboard Handler

Figure A-5 illustrates the task interfaces and data flow for RFC.

The priority of the user tasks is established by the relative position of the user task in the PDSS task queue. Those user tasks with the lower task numbers have higher priority. Therefore, USR20 has highest priority of the PDSS/IMC RFC software user tasks. A task retains control of the processor until the task releases the processor. User tasks

release control of the processor by entering a wait state (time or event).

Each of the User tasks will be defined in the following sections.

2.1 TASK EXEC

The EXEC task is the Reflight Software executive function. The EXEC task performs initialization, task communication, and termination functions for Reflight Certification.

```
TASK=EXEC
  close IMC.LOG logical unit (1)
  open IMC.LOG logical unit (1)
  IF open error then print error message and stop
  create memory region and window
  IF create error then print error message and stop
  EXEC-1:
    clear program data from ABEGIN to AEND
    activate tasks |-TASKS="FF"-|
    perform initialize setup
    initialize SEID |-SINIT( )-|
    initialize CAMAC |-CINIT( )-|
    initialize Display |-DINIT( )-|
    print message "INIT COMPLETE"
    DO UNTIL
      IF system reset then go to EXEC-1
      WAIT time |-QT20=1.0-|
    END-DO |-WAITM(QT23=1.0)-|
  END-TASK
```

2.2 TASK COMTRK

The COMTRK task is a cyclic task that executes every 0.10 seconds. This task sends the comet track serial messages to the IMCE when the comet track mode is active. The comet track data format is:

F000 F008 dddd dddd dddd dddd dddd dddd ssss

The comet track data is initially set to zero. The user may use the =MOD command to change the comet track data.

```
ASK=COMTRK
  DO UNTIL
    IF task not active |-bit #200 of TASKS-|
      then wait on task event |-WAIT(EVT21)-|
    build comet track data
    write comet track data
    wait time |-WAITM(QT21)=1.0-|
  END-DO
END-TASK
```

2.3 TASK CREW

The CREW task is a cyclic task that executes every 1.0 seconds. This task monitors the flags that indicate that an Item Entry, PFK, CMD, or TYPE command has been sensed by the Keyboard Handler. The task inserts the entry on the Scratch Pad Line and then services the crew action including the error processing associated with the action.

```
task=crew
  IF task not active |-bit #40 of tasks-|
    then wait on event |-WAIT(EVT23)-|
  CREW-1:
    IF item entry flag |-FLITEM .NE. 0-|
      then
        clear item entry flag |-FLITEM=0-|
        move item to SPL |-WDDU(ITEML,L23)-|
        clear line 19 |-WDDU(L1900,L19)-|
        IF valid item entry |-0 .GT. ITEM .LT. 23-|
          then
            perform item entry function
          else
            write error message to Line 19
              |-WDDU(L1903,L19)-|
        END-IF
    END-IF
    IF PFK flag on |-FLPFK .NE. 0-|
      then
    END-IF
    IF CMD flag on |-FLCMD .NE. 0-|
      then
        clear CMD flag |-FLCMD=0-|
        move CMD to SPL |-WDDU(ITEM1,L23)-|
        clear line 19 |-WDDU(L1900,L19)-|
        lookup sid in table
        IF valid sid and ("WRI" or "ISS")
          then
            perform CMD routine
          else
            write error message to Line 19
              |-WDDU(L1903,L19)-|
        END-IF
      END-IF
    END-IF
 END-TASK
```

2.4 TASK FLTDIS

The FLTDIS task is a cyclic task that executes every 1.0 seconds. The task updates the IMCS simulated flight crew display pages.

```
TASK=FLTDIS
  DO UNTIL
    IF task not active |-bit #20 of TASKS-|
      then WAIT on task event |-WAIT(EVT24)-|
    select display address
    update crew page |-UPAGE( )-|
    wait time |-WAITM(QT23=1.0)-|
  END-DO
END-TASK
```

2.5 TASK EXMON

The EXMON task is a cyclic task that executes every 1.0 seconds. The task performs exception monitoring on the ECIO data.

```
TASK=EXMON
  DO UNTIL
    IF task not active |-bit #20 of TASKS-|
      then wait task event |-WAIT(EVT24)-|
    convert ECIO raw analog to voltage |-SPSANL -> CAI-|
    convert ECIO analog to engineering units |-SPSANL -> KAI-|
    perform limit tests on engineering units
    compute earth's rate computations
    wait 1.0 seconds |-WAIT(QT24=1.0)-|
  END-DO
END-TASK
```

2.6 TASK ECAS

The ECAS task is a cyclic task that executes every 1.0 seconds. The task performs IMCS ECAS functions.

```
TASK=ECAS
  DO UNTIL
    IF task not active |-bit #10 of TASKS-|
      then wait task event |-WAIT(EVT25)-|
    |-Dump logic-|
    IF dump activated |-bit 3 of ECASD1-|
      then
        IF IMCE dump active |-bit 0 DI word 1-|
          then set dump started flag |-DUMPL-|
          else
            IF dump started flag on
              then
                clear dump addresses
                clear dump started flag
                reset dump selected DI
            END-IF
        END-IF
    END-IF
    |-AST VERTICAL and HORIZONTAL INTEGERS-|
    Build AST internal integers |-ECASI1 to ECAS16-|
    Wait time |-WAIT(QT25=1.0)-|
    perform table lookup for NEA
  END-DO
END-TASK
```

2.7 TASK TLOGER

The LOG task is a cyclic task that executes every 1.0 seconds and writes PDSS/IMC data to the hard disk. The task is activated/deactivated by the =LOG command which toggles the log activation function.

```
TASK=LOG
  DO UNTIL
    IF log active
      then
        WRITE buffer to disk
        IF error on WRITE
          then
            CLOSE log file
            set log inactive
          else
            increment log block count
            WAIT 1.0 seconds
        ENDIF
      else
        WAIT on LOG start event
    END-IF
  END-DO
END-TASK
```

2.8 TASK QTDISP

The QTDISP task updates the IMCS display pages per the display definition tables. The task executes every 1.0 seconds. Each execution, one display page is updated in a round robin method.

```
TASK=QTDISP
  IF task not active |-bit 1 of TASKS-|
    then wait on task event |-WAIT(EVT27)-|
  move PDSS GMT for display |-PVTIME to IMCGMT-|
  DO UNTIL
    wait 1.0 second |-WAITM(QT27=1.0)-|
  IF view active |-AVIEWD.NE.0-|
      then update view page |-UPAGE-|
          exit to END-DO
    END-IF
    DO FOR all page indexes
      get next page index |-PAGEX-|
    IF page not frozen
      then
        DO UNTIL display page completed
          get page display entry
          perform display function
        END-DO
        exit to END-DO
      END-IF
    END-DO
  END-DO
END-TASK
```

2.9 <u>TASK=QTKB</u>

The QTKB task is activated by PDSS when a keyboard entry beginning with an "=" character is detected.

```
TASK=QTKB
  DO UNTIL
    select user buffer |-USRINP=A(INBUFF)-|
    wait on keyboard event |-WAIT(EVTINP)-|
```

```
    parse out first field |-AFIELD(IXX)-| .
    look up command in table |-CMDTAB-|
    IF command found
      then perform command routine
      else print error message
    END-IF
  END-DO
END-TASK
```

3.0 ROUTINES

The PDSS/IMC routines are listed below and described in the following sections.

1	AFIELD	Parses keyboard character string
2	CCAMAC	Performs CAMAC control operation
3	CINIT	Initializes CAMAC
4	DIFET	Fetches background from disk
5	DIMOVE	Moves Display page data
6	DINIT	Initializes Displays
7	DEPERR	Generates DEP Error Message Line
8	FCHEX	Converts character string to hex
9	FCINT	Converts character string to integer
10	FCLEAR	Clears a character string
11	FCOCT	Converts character string to octal
12	FFLT	converts floating point of character string
13	FINT	Converts integer to character string
14	FINTK	Converts integer to character string (+/-)
15	FHEX	Converts integer to hex character string
16	FOCT	Converts integer to octal character string
17	FPAGE	Generates dump display page
18	IGYRO	Interrupt service for Gyro 1
19	IPDSS	Execute PDSS command
20	IRAUI	Interrupt service for RAUI
21	IRIUI	Interrupt service for RIUI 1
22	JGYRO	Interrupt service for Gyro 2
23	JRIUI	Interrupt service for RIUI
24	KGYRO	Interrupt service for Gyro 3
25	NOINT	Interrupt service for no interrupt
26	PGMT	Services =PGMT keyboard entry
27	PUTSPL	Puts messages in DDU SPL
28	QTSYSV	System verify function
29	RCAMAC	Reads CAMAC
30	RCMD	Services =C keyboard entry

31	RCOMM	Services =COMM keyboard entry
32	RCTRL	Services =CTRL keyboard entry
33	RDISP	Services =DISP keyboard entry
34	RITEM	Services =I keyboard entry
35	RLOG	Services =LOG keyboard entry
36	RMOD	Services =RMOD keyboard entry
37	RPFK	Services =P keyboard entry
38	RPGMT	Services =PGMT keyboard entry
39	RPEME	Services =PMEM keyboard entry
40	RSRST	Services =SRST keyboard entry
41	RSTAR	Services =STAR keyboard entry
42	RSTOP	Services =STOP keyboard entry
43	RTASK	Services =TASK keyboard entry
44	RTMC	Services =TMC keyboard entry
45	RTYPE	Services =T keyboard entry
46	RXPGMT	Extracts =PGMT parameters
47	RVIEW	Services =VIEW keyboard command
48	SINIT	Initializes SEID
49	UPAGE	Updates DDU flight page
50	UNINT	Interrupt service for error interrupt
51	WCAMAC	Writes to CAMAC
52	WDDU	Writes line to DDU line
53	WPDO	Writes pulsed SEID DO
54	WSDO	Writes SEID DO
55	WSPSME	Writes SPSME DO
56	WSSER	Writes Serial PCM message

3.1 ROUTINE AFIELD

The routine AFIELD parses the keyboard input character string. The routine looks for the characters < >, </>, <,>, <=>, and <CR> as separators. The parsed character string is left justified in the character string DFIELD.

Input:

IXX = address in character string where parse to begin

Output:

IXX = address of next character

DFIELD = character substring left justified

```
ROUTINE=AFIELD
  push registers onto stack
  blank DFIELD
  set # characters in DFIELD = 0
  DO UNTIL <CR> or separator found
    CASE character in input string
      <CR>:
        Exit
      <=>:
        advance to next character
      < >:
      </>:
      <,>:
        IF any characters in DFIELD
          then EXIT
        IF </>
          then
            move character to DFIELD
            increment DFIELD character count
          else
            advance to next character in input string
        END-IF
      <else>:
        move character to DFIELD
        increment DFIELD character count
        advance to next character in input string
    END-CASE
```

```
  END-DO
  save address in IXX
  pop registers from stack
END-ROUTINE
```

3.2 ROUTINE CCAMAC

The CAMAC routine performs a CAMAC control operation. CAMAC IO is performed by moving the data into the CAMAC IO address defined by the N and A codes. The CAMAC IO is a memory mapped IO.

```
Input:
     R1 = CAMAC F code
     R2 = BASE+32*N+2*A

ROUTINE=CCAMAC
     set bit #4000 in CSR |-CAMAC no read-|
     move F into CAMAC IO port |-R2-|
     clear bit #4000 in CSR |-CAMAC read-|
END-ROUTINE
```

3.3 ROUTINE CINIT

The CINIT routine initializes the CAMAC IO memory within the LSI 11/23. The following addresses have been set for CAMAC.

```
CAMAC = 170000 - CAMAC Base Address
CINT  = 170002 - CAMAC Initialize Dataway
CCLR  = 170004 - CAMAC Clear Dataway
CCSR  = 171400 - CAMAC Command Store Register
rCHDR  = 171402 - CAMAC High Data Registers
```

```
CLLR  = 171404 - CAMAC Low Register
CVCT  = 171416 - CAMAC Interrupt Vectors
```

The interrupt vectors for CAMAC are set as follows:

BASE OFFSET	ROUTINE	LAM
40	KGYRO	8
34	JGYRO	7
30	IGRYO	6
20	IRIUI	4
14	JRIUI	3
10	IRAUI	2
4	NOINT	
0	UNINT	

```
ROUTINE=CINIT
  push registers on stack
  perform dataway initialize |-CINT=0-|
  clear dataway |-CCLR=0-|
  clear CSR |-CCSR=0-|
  |-Clear LAM registers on CAMAC boards-|
  set F=0 |-CCSR(0..2)=0-|
  set no read |-CCSR(11)=1-|
  CN6A0=10. |-Gyro 1 LAM 1 reset-|
  CN6A1=10. |-Gyro 1 LAM 2 reset-|
  CN7A0=10. |-Gyro 2 LAM 1 reset-|
  CN7A1=10. |-Gyro 2 LAM 2 reset-|
  CN8A0=10. |-Gyro 3 LAM 1 reset-|
  CN8A1=10. |-Gyro 3 LAM 2 reset-|
  clear no read |-CCSR(11)=0-|
  read CN4A0 |-Read and reset RIUI 1 LAM 1-|
  read CN4A1 |-Read and reset RIUI 1 LAM 2-|
  read CN3A0 |-Read and reset RIUI 2 LAM 1-|
```

```
  read CN3A1 |-Read and reset RIUI 2 LAM 2-|
  CCSR = F(3) |-Clear RAUI-|
  CN2A13 = 6
  initialize interrupt vectors for CAMAC
  initialize CCSR
  pop registers from stack
END-ROUTINE
```

3.4 <u>ROUTINE DIFET</u>

The DIFET routine fetches background information from the disk and maps to the VRAQ or extended memory. The background information is generated by using the standard DEC Editor. The files are resident on the disk under the filenames D.xxx where xxx = 001 to 005. The background information is moved into the display buffer and blanked filled.

Input:

R2 = address of Display Table Entry

R3 = address of VRAQ or Extended Memory

```
ROUTINE=DIFET
  push registers on stack
  close logical unit 0
  lookup filename
  IF lookup error
    then
      print error message
      exit routine
  END-IF
  DO UNTIL display memory filled
```

```
    read display background file data
    IF read error
      then
        print error message
        exit routine
    END-IF
    DO for line = 1 to 80
      get character from input buffer
      IF character = <CR>
        then
          pad line with blanks
        else
          move character into display
        END-IF
      END-DO
   END-UNTIL
   pop registers from stack
END-ROUTINE
```

3.5 ROUTINE DIMOVE

The DIMOVE routine moves display pages from VRAQ memory to extended memory or visa versa.

Input:

```
     R1 = from-address
     R2 = to-address
```

```
ROUTINE=DIMOVE
  push registers on stack
  DO FOR index = 1 to 1920
    move from-address (index) to-address (index)
```

```
  END-DO
  pop registers from stack
END-ROUTINE
```

3.6 ROUTINE DINIT

The DINIT routine initializes the IMCS display pages. The background information is read in from disk and placed in VRAQ memory or in extended memory.

```
ROUTINE=DINIT
  push registers on stack
  DO UNTIL display table exhausted
    IF last entry then EXIT
    clear locator for page
    clear freeze flag
    move background from disk |-DIFET-|
  END-DO
  set flight page to no freeze
  close logical unit #0
  pop registers from stack
END-ROUTINE
```

3.7 ROUTINE DEPERR

The routine DEPERR generates the DEP error message line for the DDU display page.

INPUT: R1 = error code (ASCII)

```
ROUTINE=DEPERR
     deposit error code in line 19 message
```

```
     move GMT to line 10 message
     move line 19 message to DDU page |-WDDU(L190X,LINE=L19)-|
END-ROUTINE
```

3.8 ROUTINE FCHEX

The FCHEX routine converts a character string into a hexadecimal integer data value.

Input:

R5 = address of character string

Output:

ANSW = hexadecimal data value

```
ROUTINE=FCHEX
  push registers on stack
  DO UNTIL character string exhausted
    get next character
    ANSW = ANSW * 16 + hex (character)
  END-DO
  pop registers from stack
END-ROUTINE
```

3.9 ROUTINE FCINT

The FCINT routine converts a character string to an integer data word.

Input:

R5 = address of character string

```
ROUTINE=FOINT
  push registers on stack
  DO UNTIL characters exhausted
    pick up character
    ANSW = ANSW * 10 + integer(character)
  END-DO
  pop registers from stack.
END-ROUTINE
```

3.10 ROUTINE FCLEAR

The FCLEAR routine clears a character string.

Input:

R5 = address of character string
R1 = number of characters

```
ROUTINE=FCLEAR
  push registers on stack
  DO FOR number of characters
    move blank in character string
  END-DO
  pop registers from stack
END-ROUTINE
```

3.11 ROUTINE FCOCT

The FCOCT routine converts a character string into an octal data value.

Input:

R5 = address of character string

Output:

ANSW = octal data value

```
ROUTINE=FCOCT
  push registers on stack
  DO UNTIL character string exhausted
    get next character
    ANSW = ANSW * 8 + oct(character)
  END-DO
  pop registers from stack
END-ROUTINE
```

3.12 ROUTINE FFLT

The FFLT routine converts a floating point number to a character string of format +0.XXXXXXE+YY

Input: R2 = deposit address
R5 = address floating point numbers

```
ROUTINE=FFLT
  insert +0.000000E+00 at deposit address
  IF number negative
    then
      complement number
      inset "-" at deposit address
  END-IF
  extract integer part
  insert integer part in deposit address
  normalize integer part
  extract fraction part
  insert fraction part in deposit address
END-ROUTINE
```

3.13 ROUTINE FINT

The FINT routine converts an integer value into a character string.

Input:
R3 = integer to be converted

Output:
FLINE = character string rjs

```
ROUTINE=FINT
  push registers on stack
  DO UNTIL integer mod 10 = 0
    integer = remainder (integer/10)
    convert quotient to character
    deposit character in string
  END-DO
  pop registers from stack
END-ROUTINE
```

3.14 ROUTINE FINTK

The FINTK routine converts an integer value into a character string with a leading + or - character.

Input:
R3 = integer to be converted

Output:
FLINE = character string rjs

```
ROUTINE=FINTK
  push registers on stack
  IF integer > 0
    then deposit + in character string
    else deposit - in character string
         complement integer
    END-IF
  DO UNTIL integer mod 10 = 0
    integer = remainder (integer/10))
    convert quotient to character
    deposit character in string
  END-DO
  pop registers from stack
END-ROUTINE
```

3.15 ROUTINE FHEX

The FHEX routine converts a hexadecimal data value to a character string.

Input:
    R3 = integer to be converted

Output:
    FLINE = character string

```
ROUTINE=FHEX
  push registers on stack
  DO UNTIL integer mod 16 = 0
    integer = remainder (integer/16)
    convert quotient to character
    deposit character in string
```

```
  END-DO
  pop registers from stack
END-ROUTINE
```

3.16 ROUTINE FOCT

The FOCT routine converts an octal integer data value to a character string.

Input:
R3 = integer to be converted

Output:
FLINE = character string

```
ROUTINE-FOCT
  push registers on stack
  DO UNTIL integer mod 8 = 0
    integer = remainder (integer/8)
    convert quotient to character
    deposit character in string
  END-DO
  pop registers from stack
END-ROUTINE
```

3.17 ROUTINE FPAGE

The FPAGE routine formats a display page for a memory dump. The display page format includes the address in octal followed by 14 data values. There are 24 display lines.

Input:

R5 = address of data

R1 = display page address

Output:

display page

```
ROUTINE=FPAGE
  push registers on stack
  DO FOR 24 lines
    convert address to octal |-FOCT(address)-|
    deposit octal string on display page
    insert ":" on display page
    insert " " on display page
    DO FOR 14 data values
      fetch data value
      convert to hex-string |-FCHEX(data)-|
      deposit hex-string on display page
    END-DO
  END-DO
    Pop registers from stack
END-ROUTINE
```

3.18 ROUTINE IGYRO

The IGYRO routine fields the LAM interrupt from the GYRO #1 card.

The GYRO cards are not used for RFC.

```
ROUTINE=IGYRO
  Return from interrupt
END-ROUTINE
```

3.19 ROUTINE IPDSS

The routine IPDSS inserts a command string in the buffer FGCMD, posts the event KB for PDSS to execute the command, and then waits 5.0 seconds.

Input:
R1 = address of command string

```
ROUTINE=IPDSS
    move command string into buffer FGCMD
    post PDSS command event |-POST(KB)-|
    wait 5.0 seconds |-WAITM(QT7=5.0)-|
END-ROUTINE
```

3.20 ROUTINE IRAUI

The IRAUI routine fields the LAM from the RAUI indicating data present.

The RAUI interrupt is not processed for RFC.

```
ROUTINE=IRAUI
  return from interrupt
END-ROUTINE
```

3.21 ROUTINE IRIUI

The IRUIU routine fields the LAM from the RIUI #1 card indicating data present.

The RIUI interrupt is not processed for RFC.

```
ROUTINE=IRIUI
  return from interrupt
END-ROUTINE
```

3.22 ROUTINE JGYRO

The JGYRO routine fields the LAM interrupt from the GYRO #2 card.

The GYRO cards are not used for RFC.

```
ROUTINE=JGYRO
  return from interrupt
END-ROUTINE
```

3.23 ROUTINE JRIUI

The JRIUI routine fields the LAM from the RIUI #2 card indicating data present.

The RIUI card is not used for RFC.

```
ROUTINE=JRIUI
  return from interrupt
END-ROUTINE
```

3.24 ROUTINE KGYRO

The KGYRO routine fields the LAM from the GYRO #3 card indicating the pulse command is complete.

The GYRO cards are not used by RFC.

```
ROUTINE=KGYRO
  return from interrupt
END-ROUTINE
```

3.25 ROUTINE NOINT

The NOINT routine fields the CAMAC no interrupt condition.

```
ROUTINE=NOINT
  return from interrupt
END-ROUTINE
```

3.26 ROUTINE PGMT

The PGMT routine is invoked by the keyboard command =PGMT. The routine decodes the parameter data (day, hours, minutes, seconds) and sets the SEID GMT to the requested value.

```
ROUTINE=PGMT
  push registers on stack
  fetch parameters |-RXPGMT-|
  convert GMT to day, milliseconds in day
```

```
  build SEID set GMT command
  write set GMT command to SEID
  pop registers from stack
END-ROUTINE
```

3.27 ROUTINE PUTSPL

The routine PUTSPL moves an item entry or command message line into the simulated DDU SPL.

```
ROUTINE=PUTSPL
    move message to SPL |-WDDU(LINE22,LINE=L22)-|
    clear line 19 |-WDDU(L1900,LINE=L19)-|
END-ROUTINE
```

3.28 ROUTINE QTSYSV

The QTSYSV routine performs the system data verification functions.

```
ROUTINE=QTSYSV
  tbd
END-ROUTINE
```

3.29 ROUTINE RCAMAC

The RCAMAC routine performs a CAMAC read operation.

```
Input:
     R1 = F code
     R2 = base+32*N+2*A
     R4 = data

ROUTINE=RCAMAC
  clear F code in CSR |-bic(#7,CSR)-|
  insert F code in CSR |-bis(F,CSR)-|
  write 0 to HDR
  write data to CAMAC address |-R2-|
END-ROUTINE
```

3.30 ROUTINE RCMD

The RCMD routine services the =C keyboard entries that simulate the CMD DDU entries.

```
ROUTINE=RCMD
  set CMD flag |-FLCMD=1-|
  clear SPL |-LINE22= -|
  buffer =C entry in LINE 22
  add "CMD" to LINE 22
  add "E" to LINE 22
END-ROUTINE
```

3.31 ROUTINE RCOMM

The RCOMM routine is invoked by the =COMM keyboard command. The routine moves the keyboard data into the log comment buffer, CLLOG.

```
ROUTINE=RCOMM
  move 16 characters to log comment buffer |-CLLOG-|
END-ROUTINE
```

3.32 ROUTINE RCTRL

The RCTRL routine services the =CTRL keyboard command. The CTRL commands are:

/V Toggle verify
/T Change time (i,time)

```
ROUTINE=RCTRL
  DO UNTIL buffer emptied
    get parameter |-AFIELD-|
    IF parameter = " " then buffer emptied
    CASE parameter
      "/V":  |-toggle verify-|
        toggle verify |-XVERF-|
      "/T":  |-set time-|
        get i parameter |-AFIELD-|
        convert to integer |-FCINT-|
        IF i .GT. MAXT then exit error
        get time |-AFIELD-|
        convert to ticks |-time/PVTICKS-|
        insert time in time data value |-QTi-|
    END-CASE
  END-DO
END-ROUTINE
```

3.33 ROUTINE RDISP

The RDISP routine services the =DISP keyboard entry. The =DISP command is used to request display of IMCS display pages on the CONRAC and to freeze or unfreeze the displays. The syntax of the command is:

=DISP/f i

The /f is optional and has the value /F to freeze the page or /U to unfreeze the page.

The i is the page id which has a range from 1 to 5.

```
ROUTINE=RDISP
  deactivate view page |-AVIEWD=0-|
  get control parameter |-AFIELD-|
  IF "/" present then get parameter page id
  convert page id to integer
  IF page id > 5 then exit error
  CASE control parameter
    "/I": |-initialize backgrounds-|
      select VRAQ or extended memory
      move background to memory |-DIFET-|
    "/F": |-freeze display page-|
      set freeze in display table
    "/U":
      set update in display table
    "else": |-switch to page id-|
      IF page id not in VRAQ
        then
          move current page to extended memory |-DIMOV-|
          move page id extended memory to VRAQ
```

```
          |-DIMOV-|
      END-IF
   END-CASE
END-ROUTINE
```

3.34 ROUTINE RITEM

The RITEM routine services the =I keyboard entries that are used to simualte ITEM ENTRY DDU entries. The =I command has the following syntax.

=I item data ... data CR

```
ROUTINE=RITEM
  push resisters on stack
  clear work buffer |-ITEML[1...48]=space-|
  insert "ITEM" in work buffer
  move keyboard character string into work buffer
  |-ITEML=INBUFF-|
  set item entry flag |-FLITEM=TRUE-|
  insert "E" in work buffer
  pop registers from stack
END-ROUTINE
```

3.35 ROUTINE RLOG

The RLOG routine services the =LOG keyboard entry. The =LOG command toggles between the log function being active and inactive. The =LOG command has the format =LOG address, number-words.

```
ROUTINE=RLOG
  IF log active |-XLOG.NE.0-|
    then
      set log active |-XLOG=1-|
      post log task event |-POST(QLOG)-|
      select address |-AOFLOG=A(GMT)-|
      select number-words |-DOFLOG=HLOGN-|
      fetch parameter |-AFIELD(DFIELD)-|
      IF parameter not null
        then
        decode address |-AOFLOG=FCOCT(DFIELD)-|
        get number-words |-AFIELD(DFIELD)-|
        decode number-words |-DOFLOG=FCINT(DFIELD)-|
      END-IF
    else
      set log inactive |-XLOG=0-|
  END-IF
END-ROUTINE
```

3.36 ROUTINE RMOD

The RMOD routine services the =MOD keyboard command. The =MOD command is used to deposit data into memory. The syntax is:

=MOD address,xxxx,...,xxxx

```
ROUTINE=RMOD
  get address parameter |-AFIELD-|
  convert address to octal |-FLOCT-|
  DO UNTIL data parameters exhausted
    get data parameter |-AFIELD-|
```

```
    IF data = " " then parameters exhausted
    convert data to hex integer |-FCHEX-|
    deposit data in address
    advance address
    IF data being viewed
      then
        compute display address
        convert data to hex string
        move hex string to display address
    END-IF
  END-DO
END-ROUTINE
```

3.37 ROUTINE RPFK

The RPFK routine services PFK entries.

No PFK functions have been defined for IMCS.

```
ROUTINE=RPFK
  null
END-ROUTINE
```

3.38 ROUTINE RPGMT

The routine RPGMT services the "=PGMT" keyboard command. The "=PGMT" command has the format "=PGMT day, hour, minute, millisecond". The RPGMT routine extracts the data parameters, translates to the SEID GMT format, and writes the GMT to the SEID.

```
ROUTINE=RPGMT
    extract parameters |-RXPGMT(0)-|
    convert to day, milliseconds format
    write GMT to SEID
END-ROUTINE
```

3.39 ROUTINE RPEME

The RPEME routine services the =PEM keyboard entry that requests a printout of the IMCS display pages. The display pages are referenced by the table PMEMX which has an entry for each of the display pages.

```
ROUTINE=RPEME
  close logical unit 0
  lookup logical unit 0 to line printer
  get parameter |-AFIELD-|
  IF parameter void
    then set all display page print true
    else
      set all display page print false
      DO UNTIL parameters exhausted
        convert parameter to integer |-FCINT-|
        IF integer < 6
          then set display page print true
        get next parameter |-AFIELD-|
      END-DO
    END-IF
    DO FOR display page entries in PMEMX
      IF display page print true
        then
          get display address
```

```
        DO FOR line = 1 to 24
          move display data to buffer
          concatenate LF
          concatenate CR
          WRITE to printer
        END-DO
      END-IF
    END-DO
END-ROUTINE
```

3.40 ROUTINE RSRST

The RSRST routine services the =SRST keyboard command that causes the IMCS to reinitialize the local data.

```
ROUTINE=RSRST
  set reset flag |-XRESET=1-|
END-ROUTINE
```

3.41 ROUTINE RSTAR

The RSTAR routine services the =RSTAR keyboard command that causes the IMCS to set the start event.

```
ROUTINE=RSTAR
  post start event |-POST(EVSTRT)-|
END-ROUTINE
```

3.42 ROUTINE RSTOP

The RSTOP routine services the =STOP keyboard entry that requests the system to stop.

```
ROUTINE=RSTOP
  close logical unit 1
  close logical unit 0
  clear CSR
  clear CLR
  set log flag off |-XLOG=0-|
  set verify flag off |-XVERF=0-|
  set stop flag on |-XSTOP=1-|
END-ROUTINE
```

3.43 ROUTINE RTASK

The RTASK routine services the =TASK keyboard command. The =TASK parameter is a hex word that defines which tasks are to be active.

=TASK tttt

tttt is a hexadecimal word where bit 15 represents task 43 and bit 0 represents task 28. Each user task in PDSS/IMC monitors the TASKS variable which is set to the tttt value to determine if the task is to be active. If the task active bit is not set, the task waits on the task event, EVTxx.

```
ROUTINE=RTASK
  push registers on stack
  get parameter |-AFIELD-|
```

```
  convert parameter to hex number |-TASKS=FCHEX(DFIELD)-|
  DO for task 43 to 28
    IF task to be active
      then
        compute event EVTxx
        post event |-POST(SVTxx)-|
    END-IF
  END-DO
  pop register from stack
END-ROUTINE
```

3.44 ROUTINE RTMC

The RTMC routine services the =TMC keyboard entry that invokes the Timed Measurement Command function.

```
ROUTINE=RTMC
  tbd
END-ROUTINE
```

3.45 ROUTINE RTYPE

The RTYPE routine services the keyboard =T entries that are used to simulate "TYPE" DDU command entries.

No TYPE entries are defined for IMCS.

```
ROUTINE=RTYPE
  null
END-ROUTINE
```

3.46 ROUTINE RXPGMT

The RXPGMT routine services the IPGMT keyboard entry that is used to set GMT for PDSS and IMCE. The syntax for the =PGMT command is:

=PGMT day,hour,minute,second

```
ROUTINE=RXPGMT
  push registers on stack
  get day parameter |-AFIELD-|
  convert day to integer |-FCINT-|
  deposit day in DGMT
  get hour parameter |-AFIELD-|
  convert hour to integer |-FCINT-|
  compute 3600*hour |-JMUL(3600,hour)-|
  get minute parameter |-AFIELD-|
  convert minute to integer |-FCINT-|
  compute 60*minute
  get second parameter |-AFIELD-|
  convert second to integer |-FCINT-|
  compute second + 60*minute
  compute second + 60*minute +3600*hour
  compute 1000*sum |-JMUL(1000,sum)-|
  deposit product in DGMT
END-ROUTINE
```

3.47 ROUTINE RVIEW

The RVIEW routine services the =VIEW keyboard entry that displays PDSS memory on the CONRAC. The =VIEW command has the syntax:

=VIEW%

The % has one of the following values:

```
/S  display address = SEID table
a   a = address of data to be displayed
Ø   display address = local data
```

```
ROUTINE=RVIEW
  get VIEW parameter |-AFIELD-|
  IF parameter is "/"
    then select SEID buffer address
    else convert to address |-FCOCT-|
  END-IF
  save address |-AVIEWD=address-|
  clear display page |-FCLEAR(AVRAQ)-|
  generate display |-FPAGE-|
END-ROUTINE
```

3.48 ROUTINE SINIT

The SINIT routine initializes the SEID.

```
ROUTINE=SINIT
  push registers on stack
  DO FOR discrete = 0 to 63
    IF discrete .NE. power discrete
      then
        generate channel #
        write SEID command
    END-IF
  END-DO
END-ROUTINE
```

3.49 ROUTINE UPAGE

The UPAGE routine updates a flight page based on the flight page data table.

```
Input:
  R1 = address of display page data table
  R2 = address of display page

ROUTINE=UPAGE
  page registers on stack
  DO UNTIL display page data table exhausted
    get data type from table
    IF entry valid |-0.GE.type.LE.10-|
      then
        CASE type
          bit:
            test for bit in data
            IF bit on
              then insert field(0)
              else insert field(1)
            END-IF
          integer:
            get bit string from data
            reposition for integer
            convert to integer string |-FINTK-|
            insert integer string
          byte:
            convert to integer string
            insert integer string
          special:
            get special address
            perform special function
          hex:
```

```
          convert data to hex string
          insert hex string
        END-CASE
        IF data exceptioned monitored
          then
            IF data.GT. upper limit or
               data.LT. lower limit
               then highlight display
        END-IF
      END-IF
    END-DO
END-ROUTINE
```

3.50 ROUTINE UNINT

The UNINT routine services the CAMAC uninterrupt.

```
ROUTINE=UNINT
  return from interrupt
END-ROUTINE
```

3.51 ROUTINE WCAMAC

The WCAMAC routine performs a CAMAC write operation.

Input:

```
R1 = F
R2 = BASE+32*N+2*A
R4 = data
```

```
ROUTINE=WCAMAC
  clear F bits in CSR |-bic(#7,CCSR)-|
  set F bits in CSR |-bis(F,CCSR)-|
  clear HDR |-HDR=0-|
  write data into CAMAC IO |-(R2)=data-|
END-ROUTINE
```

3.52 ROUTINE WDDU

The WDDU routine moves a character string to the simulated DDU page for DDU lines 19-23. The simulated DDU page is located in VRAQ memory or extended memory.

Input:

R1 = address of character string

LINE = offset for line

```
ROUTINE=WDDU
  push registers on stack
  IF DDU page active
    then
      compute display address |-VRAQ+offset-|
      move input character string into display address
      pad display line
  END-IF
  pop registers from stack
END-ROUTINE
```

3.53 ROUTINE WPDO

The WPDO routine builds and issues a command to the SEID to issue a pulsed discrete output. The SEID coded mode is used.

The syntax for the SEID command where XX is the discrete output channel and NN is the on/off indicator is:

04 XX NN CR

Input:

R1 = discrete output number in ASCII
R2 = "00" for off or "01" for on

```
ROUTINE=WPDO
  push registers on stack
  move "04" in output buffer
  move XX in output buffer
  move NN in output buffer
  move CR in output buffer
  write buffer to SEID
  pop registers from stack
END-ROUTINE
```

3.54 ROUTINE WSDO

The WSDO routine builds and writes a command to the SEID to set a SEID Discrete Output. The SEID coded mode is used. The syntax of the command where XX is the discrete output channel and NN is 00 for off or 01 for on is:

02 XX NN CR

Input:

R1 = discrete output number in ASCII
R2 = "00" for off or "01" for on

```
ROUTINE=WSDO
  push registers on stack
  move "02" in output buffer
  move XX in output buffer
  move NN in output buffer
  move CR in output buffer
  write buffer to SEID
  pop registers from stack
END-ROUTINE
```

3.55 ROUTINE WSPSME

The WSPSME routine writes the command to the SEID to issue an SPSME Discrete Output. The SEID coded mode is used. The syntax of the SEID command where XX is the SPSME Discrete Output to be set is:

3A XX 01 CR

Input:

R1 = SPSME DO number in ASCII

```
ROUTINE=WSPSME
  push register on stack
  move "3A' into output buffer
  move XX into output buffer
  move "01" into output buffer
  move CR into output buffer
  write buffer to SEID
  pop registers from stack
END-ROUTINE
```

3.56 ROUTINE WSSER

The WSSER routine builds and writes a serial message to the SEID to be issued on a serial channel. The SEID coded mode is used. The syntax of the SEID message is:

OC OO NN XX ... XX CR

OO = SEID Channel 0
NN = number of words
XX = data words (max of 32)

Input:
R1 = address of data
R2 = number of data words

```
ROUTINE=WSSER
  push registers on stack
  move "OC" in output buffer
  move "OO" in output buffer
  convert number of words to hex |-NN=FHEX(R2)-|
  move NN in output buffer
  DO FOR number of data words
    convert data into hex
    move data into output buffer
  END-DO
  WRITE output buffer to SEID
END-ROUTINE
```

4.0 DATA DEFINITION

This section defines the Reflight Certification software data.

4.1 DATA TYPES

The RFC data types are listed below. The RFC software is written in Assembly Language on an LSI 11/23 microprocessor. The data types are relative to that micro which has a 16 bit basic word.

WFLOAT
- 32 bits
- DEC floating point

WINTEGER
- 16 bits
- 2's complement
- positioned on LSI 11/23 word boundary
- MSB = bit 15
- LSB = bit 0

BINTEGER
- 8 bits
- 2's complement
- positioned on LSI 11/23 byte boundary
- MSB = bit 7
- LSB = bit 0

BOOLEAN
- 8 bits
- positioned on LSI 11/23 byte boundary
- TRUE <> 0
- FALSE = 0

WBIT-STRING

16 bits

positioned on LSI 11/23 word boundary

MSB = bit 15

LSB = bit 0

BBIT-String

8 bits

positioned on LSI 11/23 byte boundary

MSB = bit 7

LSB = but 0

CHARACTER

ASCII

8 bits

ADDRESS

16 bits

positioned on LSI 11/23 word boundary

LSI address

4.2 VARIABLE DATA DEFINITIONS

ABEGIN	Pointer to start of local data area type=ADDRESS
AEND	Pointer to end of local data area type=ADDRESS
ANRXA	DRIRU computer drift data type=WFLOAT
ANRXB	DRIRU computer drift data type=WFLOAT
ANRYB	DRIRU computer drift data type=WFLOAT
ANRYC	DRIRU computer drift data type=WFLOAT
ANRZA	DRIRU computer drift data type=WFLOAT
ANRZC	DRIRU computer drift data type=WFLOAT
ASTSI	AST Serial Input Buffer type=array[1...33] of WINTEGER ASTSI[1]=number of words ASTSI[2...33]=data

AVIEWD VIEW address
type=ADDRESS
<>0 -->address and view active
=0 -->view not active

CAI Converted SPSME analog inputs
type=array[1...32] of WINTEGER
F()=20/1024(AI)

CLLOG LOG comment line
type=array[1...16] of CHARACTER
=COMM command deposit buffer

CTRACK Comet track data
type=array[1...8] of WINTEGER

DGMT RFC GMT
type=array[1...3] of WINTEGER
(1) --> GMT day
(2-3) --> milliseconds in day

DUMPB Dump start address from Item Entry 19
type=WINTEGER

DUMPC Dump start address (second part)
type=WINTEGER

DUMPE Dump length from Item Entry 20
type=WINTEGER

DUMPL Dump of AST/DEP/PCC happened flag
type=BOOLEAN
TRUE = Dump happened
FALSE = Dump not happened

DUMPX	Dump code	
	type=WINTEGER	
	X0000	No dump selected
	XF004	AST dump
	XF006	PCC dump

ECASD1	RFC ECAS Discrete Word 1	
	type=WBIT-STRING	
	Bit	
	15	HTRS Enabled
	14	IMCE Load
	13	Self Test
	12	AST Power
	5	IMCE Load
	4	Comet Track
	3	Dump Execute
	2	Dump AST
	1	Dump DEP
	0	Dump PCC

ECASD2	RFC ECAS Discrete Word 2
	type=WBIT-STRING

ECASD3	RFC ECAS Discrete Word 3
	type=WBIT-STRING

ECASI1	RFC ECAS Integer Word 1
	type=WINTEGER
	Star #1 Vertical Coordinate

ECASI2	RFC ECAS Integer Word 2
	type=WINTEGER
	Star #1 Horizontal Coordinate

ECASI3	RFC ECAS Integer Word 3 type=WINTEGER Star #2 Vertical Coordinate
ECASI4	RFC ECAS Integer Word 4 type=WINTEGER Star #2 Horizontal Coordinate
ECASI5	RFC ECAS Integer Word 5 type=WINTEGER Star #3 Vertical Coordinate
ECASI6	RFC ECAS Integer Word 6 type =WINTEGER Star #3 Horizontal Coordinate
ECASI7	RFC ECAS Integer Word 7 type=WINTEGER General Command Word #1
ECASI8	RFC ECAS Integer Word 8 type=WINTEGER General Command Word #2
ECASI9	RFC ECAS Integer Word 9 type=WINTEGER General Command Word #3
ECASIA	RFC ECAS Integer Word 10 type=WINTEGER Star #1 Brightness
ECASIB	RFC ECAS Integer Word 11 type=WINTEGER Star #2 Brightness

ECASIC	RFC ECAS Integer Word 12 type=WINTEGER Star #3 Brightness
ECASID	RFC ECAS Integer Word 13 type=WINTEGER AST Integration Time
ECASIE	RFC ECAS Integer Word 14 type=WINTEGER
ECASIF	RFC ECAS Integer Word 15 type=WINTEGER
ECASV1	RFC ECAS Variable Word 1 type=WINTEGER
ECASV2	RFC ECAS Variable Word 2 type=WINTEGER
ECASV3	RFC ECAS Variable Word 3 type=WINTEGER
ECASV4	RFC ECAS Variable Word 4 type=WINTEGER IMCE Temperature Engineering Units
ECASV5	RFC ECAS Variable Word 5 type=WINTEGER
ECASV6	RFC ECAS Variable Word 6 type=WINTEGER
ECASV7	RFC ECAS Variable Word 7 type=WINTEGER

ECASV8 RFC ECAS Variable Word 8
type=WINTEGER

ECASV9 RFC ECAS Variable Word 9
type=WINTEGER

ECASFA RFC ECAS Float Point Word 10
type=WFLOAT
Star #1 NEA

ECASFB RFC ECAS Float Point Word 11
type=WFLOAT
Star #2 NEA

ECASFC RFC ECAS Float Point Word 12
type=WFLOAT
Star #3 NEA

EDRIFT Earth Drift Data
type=WFLOAT

EMODE Earth Rate Computation Mode
type=WINTEGER

0 = inactive
1 = freeze
2 = run

EXAI Exception Monitor Flag
type=array[1...32] of Boolean

FLCMD CMD Flag
type=BOOLEAN
TRUE = CMD in queue
FALSE = No CMD in queue

FLITEM Item Entry Flag
type=BOOLEAN
TRUE = Item Entry in queue
FALSE = No Item Entry in queue

FLPFK PFK Flag
type=BOOLEAN
TRUE = PFK in queue
FALSE = No PFK in queue

FLTPGE Flight Page Flip-flop
type=WINTEGER
=0 --> Update page 1
<> --> Update page 2

FLTYPE TYPE Flag
type=BOOLEAN
TRUE = TYPE in queue
FALSE = No TYPE in queue

GYROF IMCE LAM Occurrence Indicators
type=WBIT-STRING

bit 0	-->	GYRO card #1	channel A
1	-->	1	B
2	-->	2	A
3	-->	2	B
4	-->	3	A
5	-->	3	B

IMCGMT	IMC GMT Time Obtained from PDSS type=array[1...12] of CHARACTER
INBUFF	RFC Keyboard Input Buffer type=array[1...180] of CHARACTER
ITEML	Item Entry/PFK/CMD/TYPE Temporary Buffer type=array[1...48] of CHARACTER
KAI	Engineering Units of SPSME AI type=array[1...32] of WINTEGER
LINE	DDU Display Line Offset type=ADDRESS
LINE22	DDU Line 22 Buffer type=array[1...24] of WINTEGER
MODE	RFC Mode type=WINTEGER
PAGEX	ID of Last Display Page Updated type=WINTEGER
RIUIC*	RIUI Counter (*=1,2,3,4) type=WINTEGER
RIUID*	RIUI Data (*=1,2,3,4) type=array[1...20] of WINTEGER
RIUIP*	RIUI Data Index (*=1,2,3,4) type=WINTEGER

SEIDDO	SEID DO Register type=array[1...4] of WBIT-STRING
TASKS	Active Tasks type=WBIT-STRING bit 0 --> Task 28 bit 1 --> 27 bit 14 --> 14 bit 15 --> 13
TEST	Test Data Word type=WINTEGER
TPAGEX	Temporary Holding for PAGEX type=WINTEGER
USRTK	User Task Temporary Stack type=array[1...24] of WINTEGER
WORK	RFC Work Buffer Used for SEID Outputs type=array[1...72] of WINTEGER
XLOG	Log Function Activate Flag type=BOOLEAN TRUE = active FALSE = inactive
XRESET	System Reset Indicator type=BOOLEAN TRUE = reset to be performed FALSE = reset not selected

XSTOP	System Stop Indicator type=BOOLEAN TRUE = stop FALSE = no stop
XVERF	Verification Function Indicator type=BOOLEAN TRUE = Verify is active FALSE = Verify not active
ZCSR	Copy of CAMAC CSR Register type=WINTEGER
ZSC	Local Status Code type=WINTEGER
ZZ1-5	RFC Internal Data type=WINTEGER

4.3 SEID BUFFER SPECIFICATIONS

The SEID sends the GML data to the PDSS host processor when changes are detected. The PDSS maintains the current data for the SEID data in the PDSS/SEID buffer. A definition of that data follows.

```
GMT       SEID GMT
          type=array[1...5] of WINTEGER
          (1) --> day
          (2) --> hour
          (3) --> minute
          (4) --> second
          (5) --> fractional second

MET       SEID MET
          type=array[1...5] of WINTEGER
          (1) --> day
          (2) --> hour
          (3) --> minute
          (4) --> second
          (5) --> fractional second

PCMO      PCM channel data
          type=array[1...4]   of   array[1...33]   of   WINTEGER

          (1,1)    --> Channel 0 Number of words
          (2,1)    -->         1
          (3,1)    -->         2
          (4,1)    -->         3
          (1,2-33) --> Channel 0 PCM data
          (2,2-33) -->         1
          (3,2-33) -->         2
          (4,2-33) -->         3
```

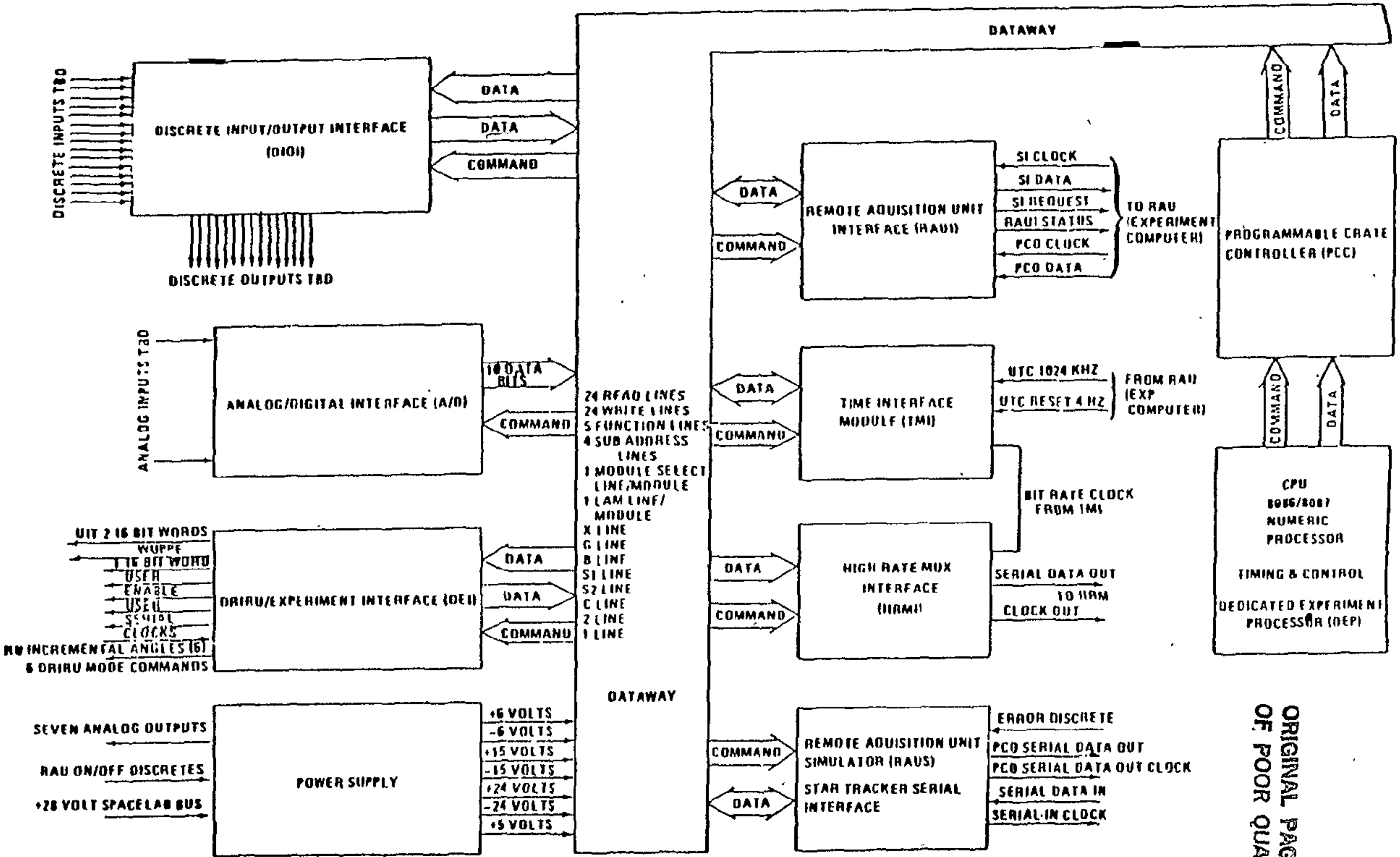
FI Flexible Inputs
type=array[1...128] of BINTEGER
8 Bit Analog

DO Discrete Output Status
type=array[1...64] of BINTEGER

SPSANL SPSME Analog Inputs
type=array[1...128] of BINTEGER
8 Bit Analog

SPSDIS SPSME Discrete Inputs
type=array[1...8] of WBIT-STRING

SPSSER SPSME Serial Input
type=array[1...33] of WINTEGER
(1) --> Number of words
(2-33) --> Serial data



ORIGINAL PAGE IS OF POOR QUALITY

FIGURE A-1: IMCE FUNCTIONAL LAYOUT WITH INTERFACES

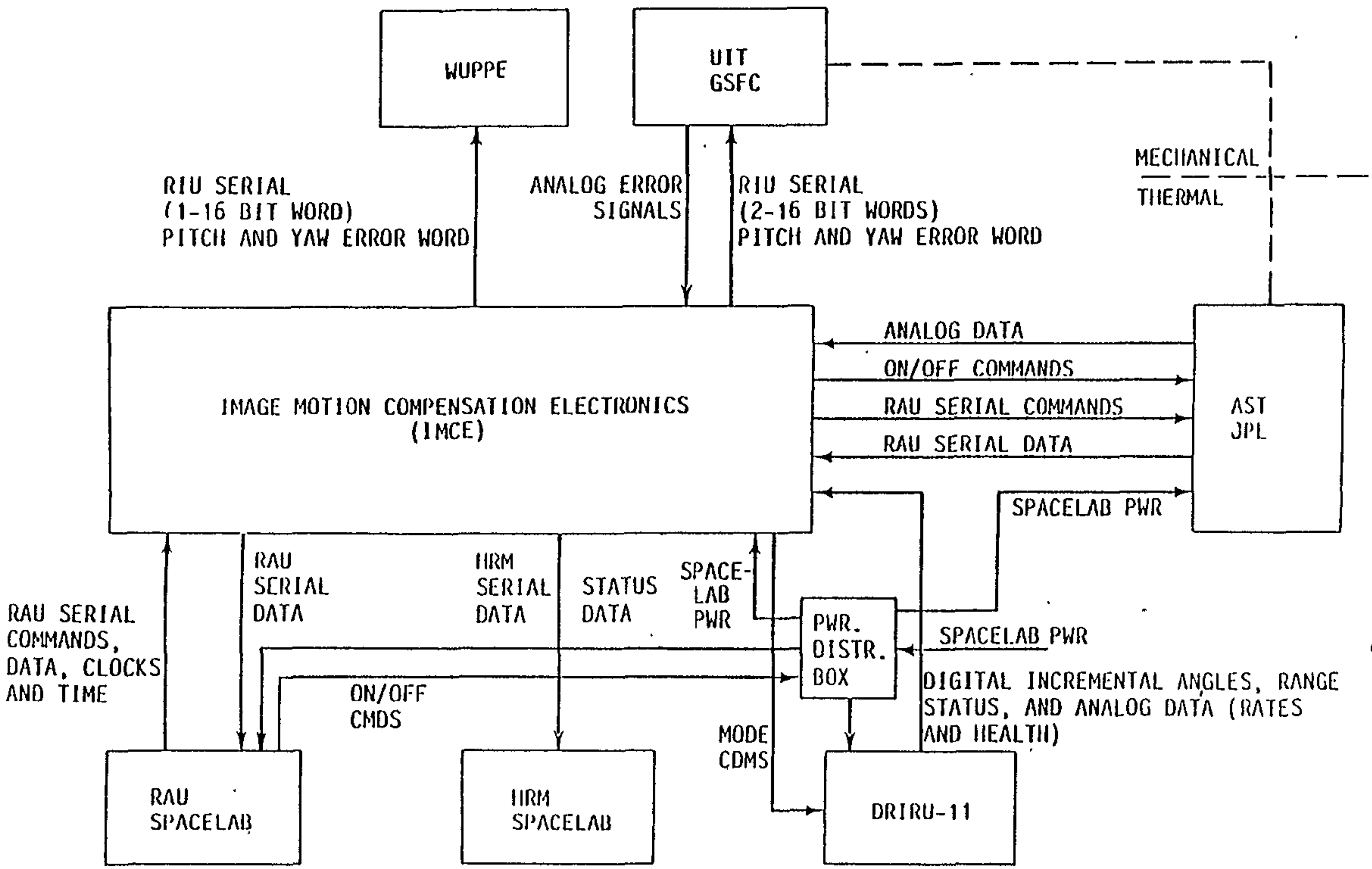
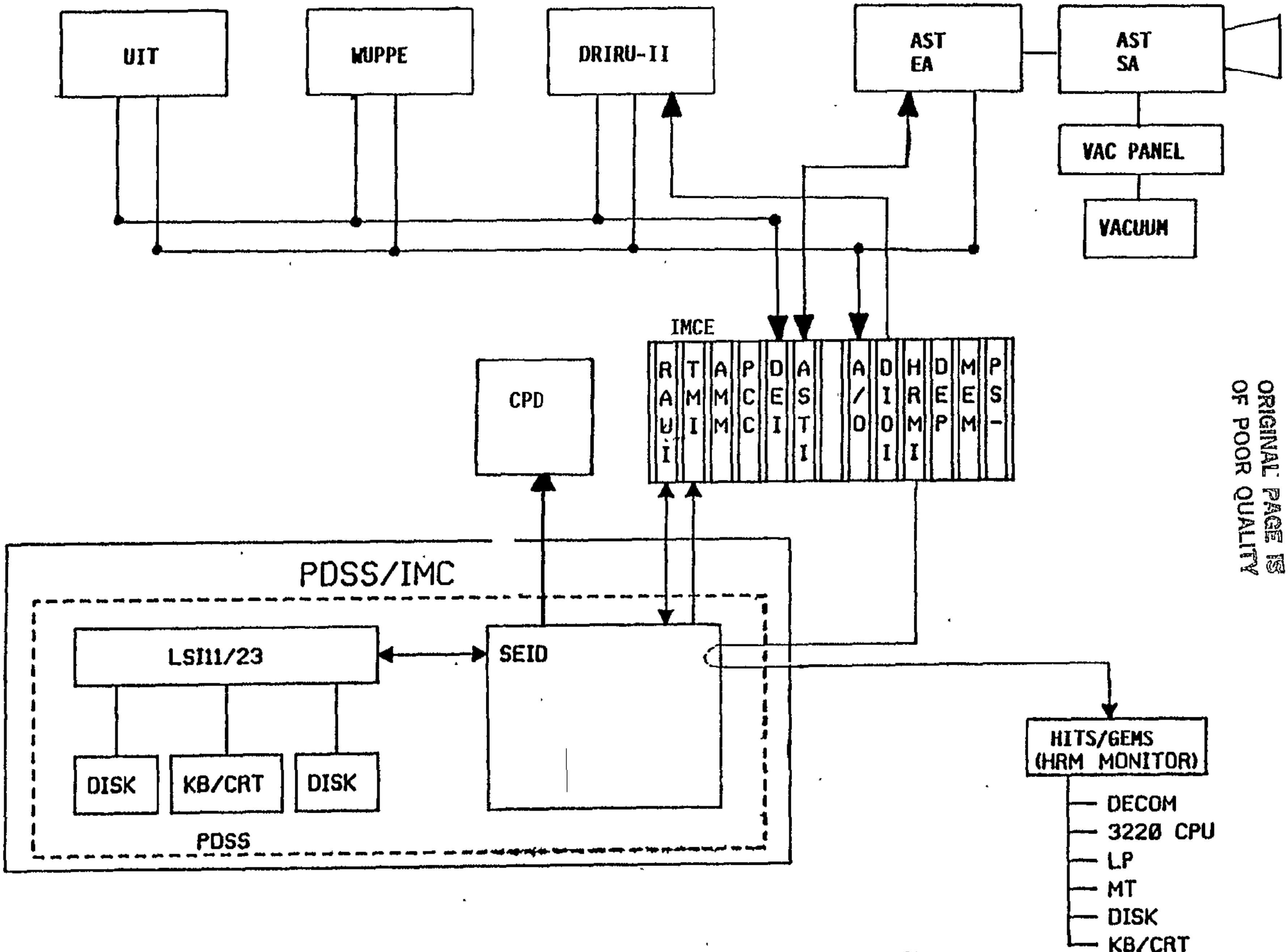


FIGURE A-2: IMAGE MOTION COMPENSATION SYSTEM (IMCS)

ORIGINAL PAGE IS
OF POOR QUALITY



ORIGINAL PAGE IS OF POOR QUALITY

FIGURE A-3: REFLIGHT CERTIFICATION CONFIGURATION

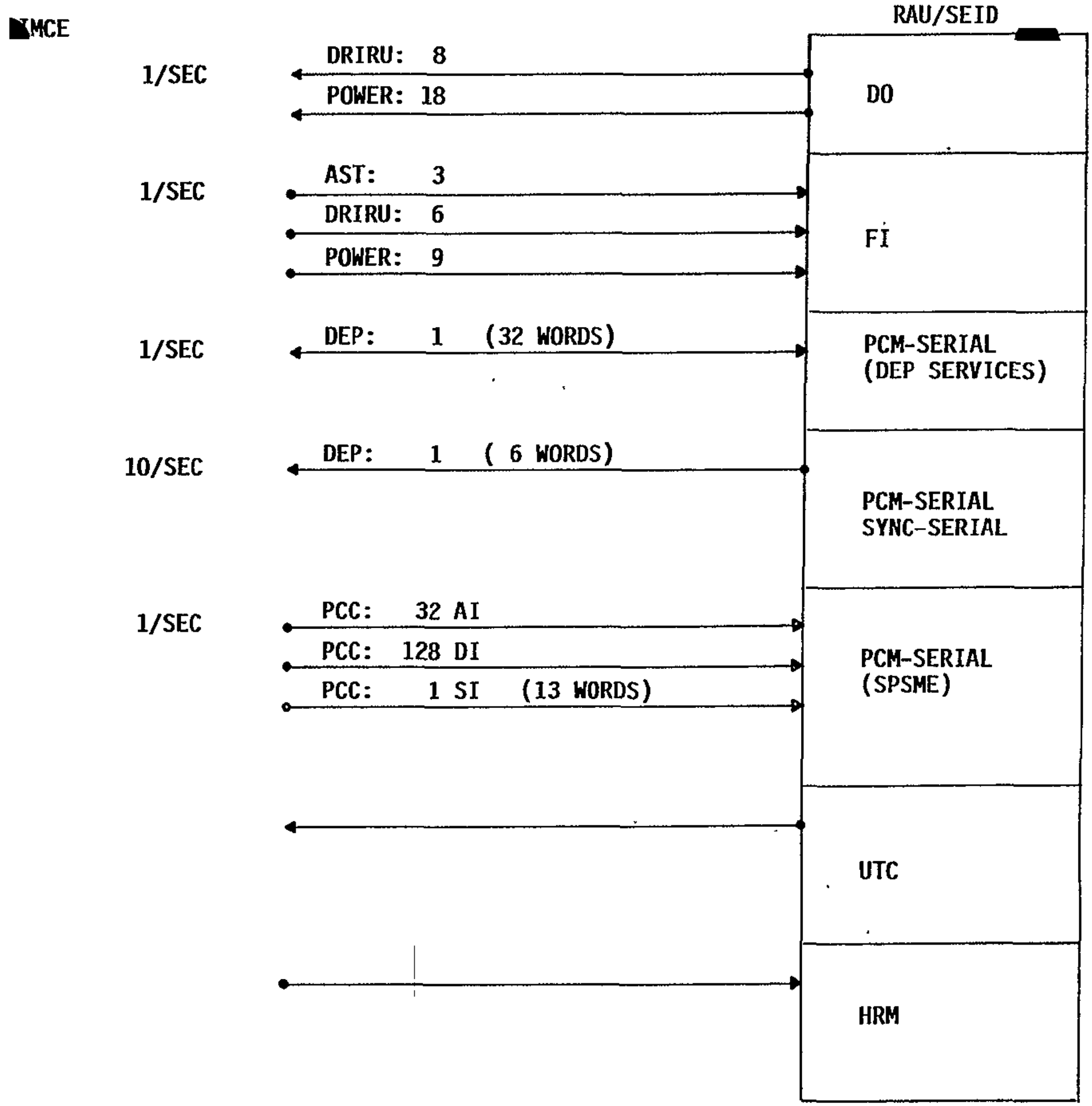


FIGURE A-4: IMCE-PDSS DATA FLOW

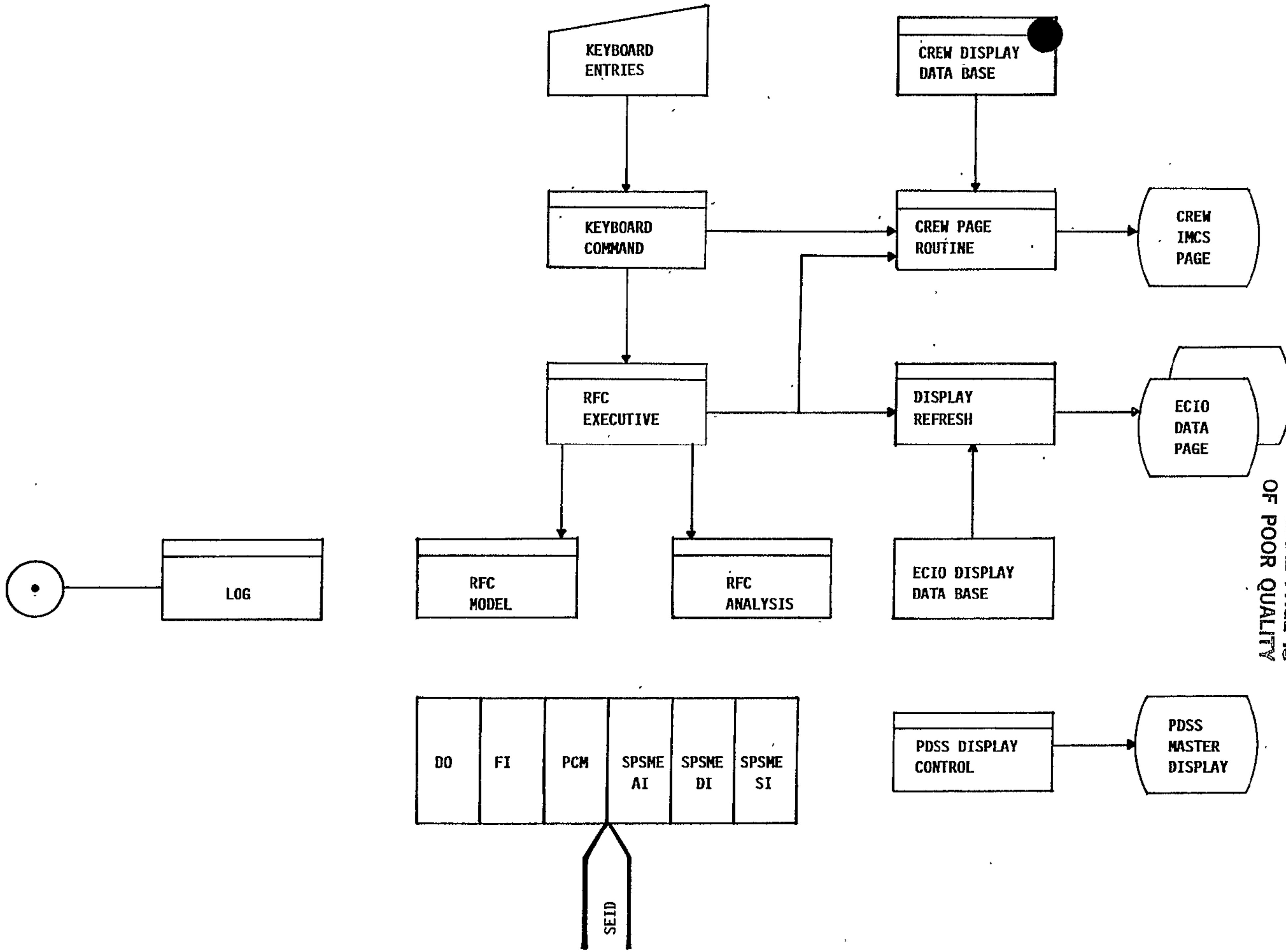


FIGURE A-5: PDSS/IMC RFC TASKS

CRT BACKGROUND FORMAT

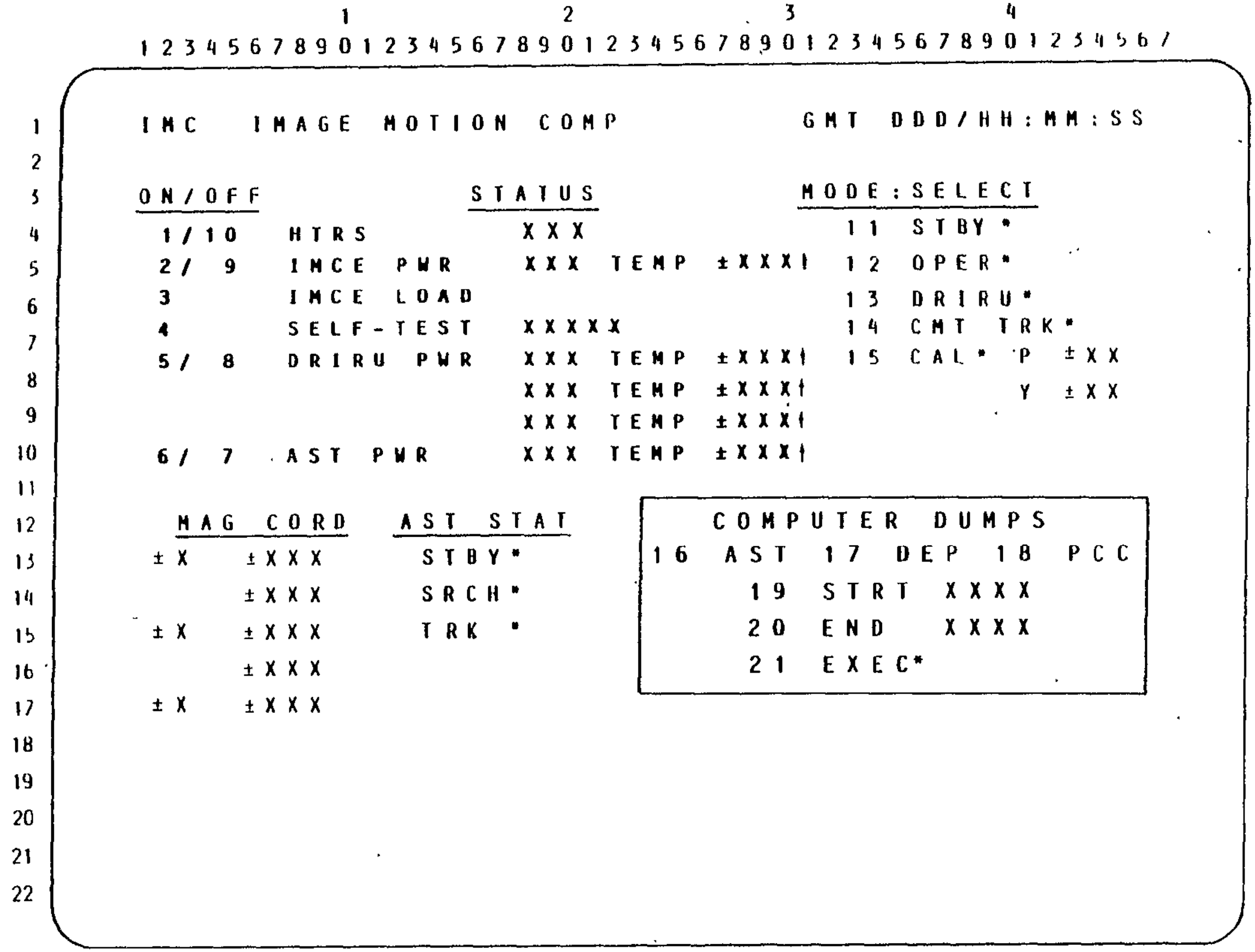


FIGURE A-6: IMCS CRT DISPLAY (TYPICAL)

FLX 00	FLX 08	FLX 16	FLX 24	FLX 32	FLX 40	FLX 48	FLX 56
FLX 01	FLX 09	FLX 17	FLX 25	FLX 33	FLX 41	FLX 49	FLX 57
FLX 02	FLX 10	FLX 18	FLX 26	FLX 34	FLX 42	FLX 50	FLX 58
FLX 03	FLX 11	FLX 19	FLX 27	FLX 35	FLX 43	FLX 51	FLX 59
FLX 04	FLX 12	FLX 20	FLX 28	FLX 36	FLX 44	FLX 52	FLX 60
FLX 05	FLX 13	FLX 21	FLX 29	FLX 37	FLX 45	FLX 53	FLX 61
FLX 06	FLX 14	FLX 22	FLX 30	FLX 38	FLX 46	FLX 54	FLX 62
FLX 07	FLX 15	FLX 23	FLX 31	FLX 39	FLX 47	FLX 55	FLX 63

PCM CHANNEL 0 LEN PAR TOT

PCM CHANNEL 1 LEN PAR TOT

PCM CHANNEL 2 LEN PAR TOT

PCM CHANNEL 3 LEN PAR TOT

PDSS TO DEP GMT: MET:

LNK USR CMD ADU DDU CDU WDU TME GNC IPS REI DSO DMG

ORIGINAL PAGE IS OF POOR QUALITY

FIGURE A-7: PDSS/IMC MASTER DISPLAY

```
 1! IMC IMAGE MOTION COMP        GMT DD/HH:MM:SS   !
 2! T_L_ID NNN H 123456  DIS DIS DIS DIS DIS DIS * !
 3!ON/OFF         STATUS         MODE:SELECT       !
 4! 1/10 HRTS       XXX            11 STBY*        !
 5! 2/ 9 IMCE PWR   XXX TEMP +XXX^ 12 OPER*        !
 6! 3    IMCE LOAD                 13 DRIRU*       !
 7! 4    SELF-TEST  XXXX           14 CMT TRK*     !
 8! 5/ 8 DRIRU PWR  XXX TEMP +XXX^ 15 CAL*         !
 9!                 XXX TEMP +XXX^                 !
10!                 XXX TEMP +XXX^ 22 MIR RESET    !
11! 6/ 7 AST PWR    XXX TEMP +XXX^                 !
12!                                FILTER SETTLED* !
13!  MAG COOD  AST STAT      COMPUTER DUMPS        !
14! +X  +XXX    STBY*    16  AST* 17  DEP* 18  PCC*!
15!     +XXX    SRCH*         19  ADDR  XXXX XXXX  !
16! +X  +XXX    TRK*          20  LNGH  XXXX       !
17!     +XXX                  21  EXEC*            !
18! +X  +XXX                                       !
19!                                                !
20!-------------------ECML-------------------------!
21!-------------------SCML-------------------------!
22!-------------------SPL--------------------------!
23!                                                !
```

FIGURE A-8: RFC DDU DISPLAY

```
 ITF IMAGE MOTION COMP         GMT DD/HH:MM:SS   !
 T_L_ID NNN H 123456  DIS DIS DIS DIS DIS DIS * !
     IMCE COMMANDS             GYRO CHANNEL XYZ  !
 3916  REBOOT            3917 A B A* 3921 B B A*  !
 3902  SELFTEST ###      3918 A B C* 3922 B B C*  !
     AST COMMANDS        3919 A C A* 3923 B C A*  !
 3925  STANDBY*          3920 A C C* 3924 B C C*  !
 3926  SEARCH*                 DRIRU CHANNEL      !
 3927  SEARCH LFOV*        01  A HIGH* 02  A LOW*!
 3928  RESET DEFECTS       03  B HIGH* 04  B LOW*!
 3929  LED ON*             05  C HIGH* 06  C LOW*!
 3930  LED OFF*          3907  DRIRU HIGH/LOW     !
 3931  LIGHT FLOOD ON*                            !
 3932  LIGHT FLOOD OFF*        AST SYNCH          !
 3933  FRAME START       3908 1HZ*  3912 3HZ*     !
  102  SET DEFECTS       3910 2HZ*  3915 4HZ*     !
  107  UPDATE INTERVAL                            !
  103  TEST COMMAND      DATA ---- ---- ----      !
                                                  !
!------------------ECML--------------------------!
!------------------SCML--------------------------!
!------------------SPL---------------------------!
                                                  !
```

FIGURE A-9: RFC DISPLAY RFC003

```
DØ4                                  IMC ECIO                           GMT=DDD,HH,MM,SS
  ANALOG BUFFER
ANRXA  +ØØØØ ANRXB  +ØØØØ ANRYB  +ØØØØ ANRYC  +ØØØØ ANRZA  +ØØØØ ANRZC  +ØØØØ
TEMPA  +ØØØØ TEMPB  +ØØØØ TEMPC  +ØØØØ T/MA   +ØØØØ T/MB   +ØØØØ T/MC   +ØØØØ
CCDTEM +ØØØØ ASTHST +ØØØØ ASTOPT +ØØØØ ASTEAT +ØØØØ ASTCPW +ØØØØ ASTH1P +ØØØØ
ASTH2P +ØØØØ ASTH3P +ØØØØ AST+5  +ØØØØ ASTBPT +ØØØØ AST+8  +ØØØØ AST+18 +ØØØØ
AST-18 +ØØØØ PSTEMP +ØØØØ PS+5   +ØØØØ PS-15  +ØØØØ PS+15  +ØØØØ ASTSAT +ØØØØ

  DISCRETE
DDDD DDDD DDDD DDDD DDDD DDDD DDDD DDDD

  SERIAL
SSSS SSSS SSSS SSSS SSSS SSSS SSSS SSSS SSSS SSSS SSSS SSSS SSSS

  E.U. ANALOG
ANRXA  +ØØØØ ANRXB  +ØØØØ ANRYB  +ØØØØ ANRYC  +ØØØØ ANRZA  +ØØØØ ANRZC  +ØØØØ
TEMPA  +ØØØØ TEMPB  +ØØØØ TEMPC  +ØØØØ T/MA   +ØØØØ T/MB   +ØØØØ T/MC   +ØØØØ
CCDTEM +ØØØØ ASTHST +ØØØØ ASTOPT +ØØØØ ASTEAT +ØØØØ ASTCPW +ØØØØ ASTH1P +ØØØØ
ASTH2P +ØØØØ ASTH3P +ØØØØ AST+5  +ØØØØ ASTBPT +ØØØØ AST+8  +ØØØØ AST+18 +ØØØØ
AST-18 +ØØØØ PSTEMP +ØØØØ PS+5   +ØØØØ PS-15  +ØØØØ PS+15  +ØØØØ ASTSAT +ØØØØ
```


FIGURE A-10: RFC DISPLAY RFC004

```
DØ5                              IMC POWER                    GMT=DDD,HH,MM,S:

   DRIRU                      AST                         IMCE
---    A POWER             ---   POWER                 ---   POWER
---    B POWER             ---   EA HEATER             ---   HEATER
---    C POWER             ---   SA HEATER

---    HEATER POWER        ---   MASTER CLOCK STATUS

+5     +15    -15    TEMP   STATUS
-----  -----  -----  -----  -----
-----  -----  -----  -----  -----
```

FIGURE A-11: RFC DISPLAY RFC005

```
DØ3                              IMCS                             GMT=DDD/HH/MM/SS

AXIS       DRIFT            EARTH DRIFT
----       -------------    -------------
XA
XB
YB
YC
ZA
ZC

<ASTROS>  SI:  ----
---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---
---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---

<WUPPE/UIT>
---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---
---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---
---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---- ---

STAR     BRIGHTNESS    INTG. TIME     NEA
#1       -----         -----          --------
#2       -----         -----          --------
#3       -----         -----          --------
```

ORIGINAL PAGE IS
OF POOR QUALITY

FIGURE A-12: AST DATA

APPENDIX B

DATA TABLES

TABLE B-1: IMCE MMU LOAD SPECIFICATIONS

ORIGINAL PAGE IS
OF POOR QUALITY

```
TITLE:  .ASCII   + MMU FILE=DEPIMC    #WORDS=369+
ELTIT:  .ASCIZ   +  +
NAME:   .ASCII   +MMUIMC+
        .EVEN
WORDS:  .WORD    FLT/FLTX+FIX-1/512.-FLTX-FIX-1
DATA:                                  ;WORD
        .WORD          0               ;( 01)   ALLOWED-FAILURES
        .WORD          0               ;( 02)   AST-TO-COUNT
        .WORD          0               ;( 03)   KDL
        .FLT2          0.0             ;( 04)   TOL-P-B
        .FLT2          0.0             ;( 06)   TOL-C-P
        .FLT2          0.0             ;( 08)   C-TOL
        .FLT2          0.0             ;( 10)   AST-BRIGHTNESS-TOL
        .FLT2          0.0             ;( 12)   AST-MOTION-TOLERANCE
        .FLT2          0.0             ;( 14)   W-CAL-AMPLITUDE
        .FLT2          2.396E+2        ;( 16)   BORE-SIGHT-COL
        .FLT2          2.901E+2        ;( 18)   BORE-SIGHT-LINE
        .FLT2          0.0             ;( 20)   UIT-MAX
        .FLT2          0.0             ;( 22)   WUPPE-MAX
        .FLT2          0.0             ;( 24)   AVERAGE-CONST
        .FLT2          0.0             ;( 26)   GRYO-NOISE[6,1]
        .FLT2          0.0             ;( 28)
        .FLT2          0.0             ;( 30)
        .FLT2          0.0             ;( 32)
        .FLT2          0.0             ;( 34)
        .FLT2          0.0             ;( 36)
        .FLT2          0.0             ;( 38)   GYRO-ACTIVE-SELECTOR[3,6]
        .FLT2          0.0             ;( 40)
        .FLT2          0.0             ;( 42)
        .FLT2          0.0             ;( 44)
        .FLT2          0.0             ;( 46)
        .FLT2          0.0             ;( 48)
        .FLT2          0.0             ;( 50)
        .FLT2          0.0             ;( 52)
        .FLT2          0.0             ;( 54)
        .FLT2          0.0             ;( 56)
        .FLT2          0.0             ;( 58)
        .FLT2          0.0             ;( 60)
        .FLT2          0.0             ;( 62)
        .FLT2          0.0             ;( 64)
        .FLT2          0.0             ;( 66)
        .FLT2          0.0             ;( 68)
        .FLT2          0.0             ;( 70)
        .FLT2          0.0             ;( 72)
        .FLT2          0.0             ;( 74)   GYRO-PRIME-SELECTOR[3,6]
        .FLT2          0.0             ;( 76)
        .FLT2          0.0             ;( 78)
        .FLT2          0.0             ;( 80)
        .FLT2          0.0             ;( 82)
        .FLT2          0.0             ;( 84)
        .FLT2          0.0             ;( 86)
        .FLT2          0.0             ;( 88)
        .FLT2          0.0             ;( 90)
        .FLT2          0.0             ;( 92)
        .FLT2          0.0             ;( 94)
        .FLT2          0.0             ;( 96)
        .FLT2          0.0             ;( 98)
        .FLT2          0.0             ;(100)
        .FLT2          0.0             ;(102)
        .FLT2          0.0             ;(104)
        .FLT2          0.0             ;(106)
        .FLT2          0.0             ;(108)
        .FLT2          0.0             ;(110)   GYRO-BACKUP-SELECTOR[3,6]
        .FLT2          0.0             ;(112)
        .FLT2          0.0             ;(114)
        .FLT2          0.0             ;(116)
```

TABLE B-1: IMCE MMU LOAD SPECIFICATIONS
(CONTINUED)

ORIGINAL PAGE IS
OF POOR QUALITY

```
.FLT2    0.0           ;(118)
.FLT2    0.0           ;(120)
.FLT2    0.0           ;(122)
.FLT2    0.0           ;(124)
.FLT2    0.0           ;(126)
.FLT2    0.0           ;(128)
.FLT2    0.0           ;(130)
.FLT2    0.0           ;(132)
.FLT2    0.0           ;(134)
.FLT2    0.0           ;(136)
.FLT2    0.0           ;(138)
.FLT2    0.0           ;(140)
.FLT2    0.0           ;(142)
.FLT2    0.0           ;(144)
.FLT2    0.0           ;(146)  WEIGHT-FACTOR[3,1]
.FLT2    0.0           ;(148)
.FLT2    0.0           ;(150)
.FLT2    0.0           ;(152)  GYRO-SCALE-FACTORS[6,3]
.FLT2    0.0           ;(154)
.FLT2    0.0           ;(156)
.FLT2    0.0           ;(158)
.FLT2    0.0           ;(160)
.FLT2    0.0           ;(162)
.FLT2    0.0           ;(164)
.FLT2    0.0           ;(166)
.FLT2    0.0           ;(168)
.FLT2    0.0           ;(170)
.FLT2    0.0           ;(172)
.FLT2    0.0           ;(174)
.FLT2    0.0           ;(176)
.FLT2    0.0           ;(178)
.FLT2    0.0           ;(180)
.FLT2    0.0           ;(182)
.FLT2    0.0           ;(184)
.FLT2    0.0           ;(186)
.FLT2    0.0           ;(188)  PRELAUNCH-DRIFT-RATES[6,3]
.FLT2    0.0           ;(190)
.FLT2    0.0           ;(192)
.FLT2    0.0           ;(194)
.FLT2    0.0           ;(196)
.FLT2    0.0           ;(198)
.FLT2    0.0           ;(200)
.FLT2    0.0           ;(202)
.FLT2    0.0           ;(204)
.FLT2    0.0           ;(206)
.FLT2    0.0           ;(208)
.FLT2    0.0           ;(210)
.FLT2    0.0           ;(212)
.FLT2    0.0           ;(214)
.FLT2    0.0           ;(216)
.FLT2    0.0           ;(218)
.FLT2    0.0           ;(220)
.FLT2    0.0           ;(222)
.FLT2    1.0868E-11    ;(224)  NEA-TABLE[31,1]
.FLT2    4.9735E-12    ;(226)
.FLT2    1.9767E-12    ;(228)
.FLT2    1.1376E-12    ;(230)
.FLT2    6.7928E-13    ;(232)
.FLT2    4.6069E-13    ;(234)
.FLT2    3.3846E-13    ;(236)
.FLT2    2.8442E-13    ;(238)
.FLT2    2.3504E-13    ;(240)
.FLT2    2.3504E-13    ;(242)
.FLT2    1.9039E-13    ;(244)
.FLT2    1.9039E-13    ;(246)
.FLT2    1.5043E-13    ;(248)
```

TABLE B-1: IMCE MMU LOAD SPECIFICATIONS
(CONTINUED)


```
.FLT2     1.5043E-13     ;(250)
.FLT2     1.1517E-13     ;(252)
.FLT2     1.1517E-13     ;(254)
.FLT2     1.1517E-13     ;(256)
.FLT2     8.4616E-14     ;(258)
.FLT2     8.4616E-14     ;(260)
.FLT2     8.4616E-14     ;(262)
.FLT2     8.4616E-14     ;(264)
.FLT2     8.4616E-14     ;(266)
.FLT2     5.8761E-14     ;(268)
.FLT2     5.8761E-14     ;(270)
.FLT2     5.8761E-14     ;(272)
.FLT2     5.8761E-14     ;(274)
.FLT2     5.8761E-14     ;(276)
.FLT2     5.8761E-14     ;(278)
.FLT2     5.8761E-14     ;(280)
.FLT2     5.8761E-14     ;(282)
.FLT2     5.8761E-14     ;(284)
.FLT2     0.0            ;(286)   P-TRANSFORM[3,3]
.FLT2     0.0            ;(288)
.FLT2     0.0            ;(290)
.FLT2     0.0            ;(292)
.FLT2     0.0            ;(294)
.FLT2     0.0            ;(296)
.FLT2     0.0            ;(298)
.FLT2     0.0            ;(300)
.FLT2     0.0            ;(302)
.FLT2     0.0            ;(304)   PA-TRANSFORM[3,3]
.FLT2     0.0            ;(306)
.FLT2     0.0            ;(308)
.FLT2     0.0            ;(310)
.FLT2     0.0            ;(312)
.FLT2     0.0            ;(314)
.FLT2     0.0            ;(316)
.FLT2     0.0            ;(318)
.FLT2     0.0            ;(320)
.FLT2     0.0            ;(322)   U-TRANSFORM[3,3]
.FLT2     0.0            ;(324)
.FLT2     0.0            ;(326)
.FLT2     0.0            ;(328)
.FLT2     0.0            ;(330)
.FLT2     0.0            ;(332)
.FLT2     0.0            ;(334)
.FLT2     0.0            ;(336)
.FLT2     0.0            ;(338)
.FLT2     0.0            ;(340)   W-TRANSFORM[3,3]
.FLT2     0.0            ;(342)
.FLT2     0.0            ;(344)
.FLT2     0.0            ;(346)
.FLT2     0.0            ;(348)
.FLT2     0.0            ;(350)
.FLT2     0.0            ;(352)
.FLT2     0.0            ;(354)
.FLT2     0.0            ;(356)
.WORD     0              ;(358)   NUMBER-DEFECT-COORDS
.WORD     0              ;(359)   DEFECT-COORDS[10,1]
.WORD     0              ;(360)
.WORD     0              ;(361)
.WORD     0              ;(362)
.WORD     0              ;(363)
.WORD     0              ;(364)
.WORD     0              ;(365)
.WORD     0              ;(366)
.WORD     0              ;(367)
.WORD     0              ;(368)
.WORD     0              ;(369)   SENT-CHECKSUM
```

TABLE B-2: ECIO ANALOG DATA

NAME	DESCRIPTION	SAMPLE RATE	SIZE (BITS)	SPSANL INDEX
	Spare	1	8	0
	Spare	1	8	1
ANRXA	X Axis Rate A	1	8	2
ANRXB	X Axis Rate B	1	8	3
ANRYB	Y Axis Rate B	1	8	4
ANRYC	Y Axis Rate C	1	8	5
ANRZA	Z Axis Rate A	1	8	6
ANRZC	Z Axis Rate C	1	8	7
TEMPA	A GYRO Temperature	1	8	8
TEMPB	B GYRO Temperature	1	8	9
TEMPC	C GYRO Temperature	1	8	10
T/MA	A GYRO Motor Current	1	8	11
T/MB	B GYRO Motor Current	1	8	12
T/MC	C GYRO Motor Current	1	8	13
ACCDT	AST CCD Temperature	1	8	14
AHST	AST Heat Sink Temperature	1	8	15
AOPT	AST Optics Temperature	1	8	16
AEAT	AST EA Temperature	1	8	17
ACCDV	AST CCD Cooler Volt	1	8	18
AH1V	AST Heater 1 Volt	1	8	19
AH2V	AST Heater 2 Volt	1	8	20
AH3V	AST Heater 3 Volt	1	8	21
AP5V	AST +5 Volts	1	8	22
ABPT	AST Baseplate Temperature	1	8	23
AP8V	AST +8 Volts	1	8	24
AP18V	AST +18 Volts	1	8	25
AN18V	AST -18 Volts	1	8	26
PSTEMP	IMCE Temperature	1	8	27
PS+5V	PS +5	1	8	28
PSN15V	PS -15 Volts	1	8	29
PSP15V	PS +15 Volts	1	8	30
ASAT	AST SA Electronics Temperature	1	8	31

Data deposited in SPSANL

TABLE B-3: ECIO DISCRETE DATA

DESCRIPTION			NUMBER OF BITS	BIT POSITION	DATA TYPE
Software Status Parent Word 1					
-Load MMU	On/Off	DEP 1-01	1	15	B
-Load OK	Y/N	DEP 1-02	1	14	B
-Test	Go/Nogo	DEP 1-03	1	13	B
-DRI Mode	Hi/Lo	DEP 1-04	1	12	B
-Standby	On/Off	DEP 1-05	1	11	B
-Operate	On/Off	DEP 1-06	1	10	B
-DRI (Only)	On/Off	DEP 1-07	1	9	B
-Mirror Reset	On/Off	DEP 1-08	1	8	B
-Comet	On/Off	DEP 1-09	1	7	B
-Calibrate	On/Off	DEP 1-10	1	6	B
-AST Standby	Y/N	DEP 1-11	1	5	B
-AST Search	Y/N	DEP 1-12	1	4	B
-AST Track	Y/N	DEP 1-13	1	3	B
-Filter Settled	Y/N	DEP 1-14	1	2	B
-IMCE Power	On/Off	DEP 1-15	1	1	B
-AST Dump	Y/N	DEP 1-16	1	0	B
DEP Software Status Parent Word 2					
-XA YB ZA		DEP 2-01	1	15	B
-XA YB ZC		DEP 2-02	1	14	B
-XA YC ZA		DEP 2-03	1	13	B
-XA YC ZC		DEP 2-04	1	12	B
-XB YB ZA		DEP 2-05	1	11	B
-XB YB ZC		DEP 2-06	1	10	B
-XB YC ZA		DEP 2-07	1	9	B
-XB YC ZB		DEP 2-08	1	8	B
-PCC Dump	On/Off	DEP 2-09	1	7	B
-Spare			7	0-6	B

TABLE B-3: ECIO DISCRETE DATA
(CONTINUED)

DESCRIPTION			NUMBER OF BITS	BIT POSITION	DATA TYPE
DEP Hardware Status Parent Word					
-1 Memory Error			1	15	B
-2 PCC Communication Error			1	14	B
-3 System Interrupt Error			1	13	B
-4 8087 Computational Error			1	12	B
-5 Running in Monitor			1	11	B
-6 Error 6			1	10	B
-7 Error 7			1	9	B
-8 Error 8			1	8	B
-9 Error 9			1	7	B
-10 Error 10			1	6	B
-11 Error 11			1	5	B
-12 Error 12			1	4	B
-13 Error 13			1	3	B
-14 Error 14			1	2	B
-15 Error 15			1	1	B
-16 error 16			1	0	B
PCC Software Statue Parent Word 1					
-Telemetry	On/Off	PCC01	1	15	B
-RAU	On/Off	PCC02	1	14	B
-Spare			11	13-3	B
-PCC Memory Test	Error/Noerr	PCC14	1	2	B
-Spare			2	1-0	B
Group 1 DI Parent Word					
-Spare			10	15-6	B
-DRI Range Status ZC DI			1	5	B
-DRI Range Statue ZA DI			1	4	B
-DRI Range Status YC DI			1	3	B
-DRI Range Status YB DI			1	2	B
-DRI Range Status XB DI			1	1	B
-DRI Range Status XB DI			1	1	B
-DRI Range Status XA DI			1	0	N

TABLE B-3: ECIO DISCRETE DATA
(CONTINUED)

DESCRIPTION	NUMBER OF BITS	BIT POSITION	DATA TYPE
DRI Mode Command Group DO's Parent Word			
-Spare	10	15-6	B
-DRI Mode Command C, Low	1	5	B
-DRI Mode Command C, High	1	4	B
-DRI Mode Command B, Low	1	3	B
-DRI Mode Command B, High	1	2	B
-DRI Mode Command A, Low	1	1	B
-DRI Mode Command A, High	1	0	B
RAUI Status Parent Word			
-Spare	10	15-6	B
-PCO Buffer Overflow	1	5	B
-RAU Did Not Take All RAUI Data	1	4	B
-PCO Data Word Parity Error	1	3	B
-STSW Parity Error	1	2	B
-Non-Valid STSW	1	1	B
-Parity Bit	1	0	B
Group 0 DI Parent Word			
-Master Clock Status	1	15	B
-Spare	15	14-0	B

Data deposited in SPSDIS

C-2

TABLE B-4: ECIO SERIAL DATA

DESCRIPTION	NUMBER OF BITS	BIT POSITION	DATA TYPE
AST Wrap Around Counter	16	0	U
AST Data Word 1 Parent	16		
-AST Update Interval (MS)	9	15-7	U
-AST Memory Dump On/Off	1	6	B
-AST Self Test Star On/Off	1	5	B
-AST Error Flag Normal/Error	1	4	B
-AST Thermoelectric Cooler Power On/Off	1	3	B
-AST Rate Flag	1	2	B
-AST Operation Mode	2	1-0	U
AST Data Word 2 Parent	16	0	N
-AST Light Flood Status	1	15	B
-AST Brightness of 1st Star	5	14-10	U
-AST Brightness of 2nd Star	5	9-5	U
-AST Brightness of 3rd Star	5	4-0	U
AST Data Word 3 Parent	16	0	N
-AST Error Number	4	15-12	N
-AST Integration Time (MS)	12	11-0	U
AST Vertical Coord. of 1st Star (16 LSB)	16	0	U
AST Horizontal Coord. of 1st Star (16 LSB)	16	0	U
AST Vertical Coord. of 2nd Star (16 LSB)	16	0	U
AST Horizontal Coord. of 2nd Star (16 LSB)	16	0	U
AST Vertical Coord. of 3rd Star (16 LSB)	16	0	U
AST Horizontal Coord. of 3rd Star (16 LSB)	16	0	U
AST Data Word 10 Parent	16	0	N
-Spare	4	15-12	
-AST Vertical Coord. of 1st Star (2 MSB)	2	11-10	U
-AST Hor. Coord. of 1st Star (2 MSB)	2	9-8	U
-AST Vertical Coord. of 2nd Star (2 MSB)	2	7-6	U
-AST Hor. Coord. of 2nd Star (2 MSB)	2	5-4	U
-AST Vertical Coord. of 3rd Star (2 MSB)	2	3-2	U
-AST Hor. Coord. of 3rd Star (2 MSB)	2	1-0	U
Calibrate Mode Y	16	0	U
Calibrate Mode Z	16	0	U

Data deposited in SPSSER

TABLE B-5: DISPLAY TYPES

TYPE	DESCRIPTION
b	bit test
i	integer
j	subinteger
h	hex
v	voltage

LOGIC

```
b - IF (DATA .and. MASK) = 1
      then bit is off
      else bit is on

i   DATA = integer

j   rjs (DATA .and. MASK)

h   hexadecimal integer

v   voltage = 20/255(DATA+.5)*100
```

TABLE B-6: IMCS CREW PAGE DISPLAY ELEMENTS

NO.	ELEMENT
1	HTRS xxx
2	IMCE PWR xxx
3	IMCE LOAD*
4	SELF TEST xxx
5	DRIRU PWR xxx
6	xxx
7	xxx
8	AST PWR xxx
9	AST TEMP +xxx
16	STBY*
17	OPER*
18	DRIRU*
19	CMT TRK*
20	CAL*
21	AST*
22	DEP*
23	PCC*
24	STRT xxxx
25	LNGH xxxx
30	IMCE TEMP +xxx
26	EXEC*
31	DRIRU TEMP +xxx
32	+xxx
33	+xxx
35	MAG CORD +x
36	+xxx
37	+xxx
38	+x
39	+xxx
40	+xxx
41	+x
42	+xxx
43	+xxx
44	AST STBY*
45	SRCH*
46	TRK*
47	FILTER SETTLED*
48	STRT ---- XXXX
49	MIRROR RESET*

TABLE B-7: FLIGHT CREW PAGE

NO.	TYPE	LN	SOURCE	DISPLAY	sid
0	v	4	KAI (27)		3279
1	b	3	ECASD1 x8000	INH ENA	
2	b	3	SPSDIS x0002	OFF ON	
3	b	1	ECASD1 x4000	*	
4	b	5	SPSDIS x2000	NOGO GO	
5	v	4	KAI (11)		3263
6	v	4	KAI (12)		3264
7	v	4	KAI (13)		3265
8	v	4	KAI (22)		3274
9	v	4	KAI (17)		3269
16	b	1	SPSDIS x0800	*	
17	b	1	SPSDIS x0400	*	
18	b	1	SPSDIS x0020	*	
19	b	1	SPSDIS x0080	*	
20	b	1	SPSDIS x0040	*	
21	b	1	ECASD1 x0004	*	
22	b	1	ECASD1 x0002	*	
23	b	1	ECASD1 x0001	*	
24	h	4	DUMPB		
25	h	4	DUMPE		
26	b	1	SPSDIS x0008	*	
31	v	4	KAI (8)		3253
32	v	4	KAI (9)		3254
33	v	4	KAI (10)		3256
35	j	2	SPSSER(6) x7C00		3286
36	i	4	ECASI1		
37	i	4	ECASI2		
38	j	2	SPSSER(6) x03E0		3287
39	i	4	ECASI3		
40	i	4	ECASI4		
41	j	2	SPSSER(6) x001F		3288
42	i	4	ECASI5		
43	i	4	ECASI6		
44	b	1	SPSDIS x0020	*	
45	b	1	SPSDIS x0010	*	
46	b	1	SPSDIS x0008	*	
47	b	1	SPSDIS x0004	*	
48	h	4	DUMPC		

TABLE B-8: EXCEPTION MONITOR

INDEX	UPPER	LOWER	CONVERSION	
1	+8.00	-8.00	0.0	.01955034
2	+8.00	-8.00	0.0	.01955034
3	+0.111	-0.111	0.0	.00032552
4	+0.111	-0.111	0.0	.00032552
5	+0.111	-0.111	0.0	.00032552
6	+0.111	-0.111	0.0	.00032552
7	+0.111	-0.111	0.0	.00032552
8	+0.111	-0.111	0.0	.00032552
9	+65.00	-10.0	84.075142	-.433526
10	+65.00	-10.0	84.075412	-.433526
11	+65.00	-10.0	84.075412	-.433526
12	+200.0	0.0	0.0	.079557026
13	+200.0	0.0	0.0	.079557026
14	+200.0	0.0	0.0	.079557026
15	-47.0	-67.0	-57.0	.02941175
16	+45.0	+15.0	+30.0	.09803925
17	30.0	10.0	20.0	.0490195
18	50.0	-10.0	20.0	.09803925
19	7.0	4.5	5.75	.00431373
20	0.0	-10.0	-5.0	.027451
21	0.0	-10.0	-5.0	.027451
22	0.0	-10.0	-5.0	.027451
23	5.25	4.75	5.0	.01117648
24	30.0	10.0	20.0	.09803925
25	10.0	7.5	8.75	.01078431

TABLE B-8: EXCEPTION MONITOR
(CONTINUED)

INDEX	UPPER	LOWER	CONVERSION	
26	20.5	17.5	19.0	.02156863
27	-20.5	-17.5	-19.0	.02156863
28	+8.00	-8.00	30.196087	.39219668
29	+8.00	-8.00	.080321	.1600274
30	+8.00	-8.00	.08065	.16129
31	+8.00	-8.00	.024113	.0482026
32	50.0	-10.0	20.0	.09803925
33	+8.00	-8.00	-5.12	.04015686
34	+8.00	-8.00	-5.12	.04015686
35	+8.00	-8.00	-5.12	.04015686
36	+8.00	-8.00	-5.12	.04015686
37	+8.00	-8.00	-5.12	.04015686

*S( ) = SPSME Analog Input
A( ) = RAU Flexible Input

TABLE B-9: ITEM ENTRIES

ITEM	FUNCTION	ACTION
1	HTRS ENA	Issue DOP - IMCE Heater On SID=#3370,DOP=11,SEID=58 Issue DOP - AST EA Heater On SID=#3374,DOP=15,SEID=62 Issue DOP - AST-SA Heater On SID=3386,DOP=27,SEID=32
2	IMCE PWR ON	Issue DOP - IMCE Power On SID=#3368,DOP=9,SEID=56
3	IMCE LOAD	DEP Protocol MMU Load
4	SELF TEST	Issue SPSME DO 31 SID=#3902,WRI=001F,SDO=31
5	DRIRU PWR ON	Issue DOP - DRIRU A Power On SID=#3360,DOP=1,SEID=48 Issue DOP - DRIRU B Power On SID=#3362,DOP=3,SEID=50 Issue DOP - DRIRU C Power On SID=#3364,DOP=5,SEID=52
6	AST PWR ON	Issue DOP - AST Power On SID=#3372,DOP=13,SEID=60
7	AST PWR OFF	Issue DOP - AST Power Off SID=#3373,DOP=14,SEID=61
8	DRIRU PWR OFF	Issue DOP - DRIRU X Power Off SID=#3361,DOP=2,SEID=49 Issue DOP - DRIRU Y Power Off SID=#3363,DOP=4,SEID=51 Issue DOP - DRIRU Z Power Off SID=#3365,DOP=6,SEID=53
9	IMCE PWR OFF	Issue DOP - IMCE Power Off SID=#3369,DOP=10,SEID=57
10	HTRS INHIBIT	Issue DOP - IMCE Heater Off SID=#3371,DOP=12,SEID=59 Issue DOP - AST EA Heater Off SID=#3375,DOP=16,SEID=63 Issue DOP - AST SA Heater Off SID=3387,DOP=28,SEID=33

TABLE B-9: ITEM ENTRIES
(CONTINUED)

ITEM	FUNCTION	ACTION
11	STBY	Issue SPSME DO - Standby SID=#3903,WRI=0001,SDO=1
12	OPER	Issue SPSME DO - Operate SID=#3904,WRI=0002,SDO=2
13	DRIRU	Issue SPSME DO - DRIRU Only SID=#3905,WRI=0003,SDO=3
14	CMTRK	Issue SPSME DO - Comet Track SID=#3909,WRI=0007,SDO=7
15	CAL	Issue SPSME DO - Calibrate SID=#3911,WRI=0009,SDO=9
16	AST DUMP	
17	DEP DUMP	
18	PCC DUMP	
19	START	Data=start address
20	LNGH	Data=length
21	EXEC	Issue Dump Serial Message SID=TBD,WRI=F00x,ssss,llll
22	MIRROR RST	Issue SPSME DO - Mirror Reset SID=3938,WRI=0030,SEID=48

TABLE B-10: GENERALIZED COMMAND (NO DATA)

:CMD: ISS-sid :ENTER:

SID	COMMAND	SDO	WRI
3907	DRIRU High/Low	5	0005
3908	AST SYNCH 1HZ	6	0006
3910	AST SYNCH 2HZ	8	0008
3912	AST SYNCH 3HZ	10	000A
3915	AST SYNCH 4HZ	12	000C
3916	REBOOT	11	000B
3917	GYRO CHNL XA,YB,ZA	13	000D
3918	XA,YB,ZC	14	000E
3919	XA,YC,ZA	16	0010
3920	XA,YC,ZC	17	0011
3921	XB,YB,ZA	18	0012
3922	XB,YB,ZC	19	0013
3923	XB,YC,ZA	20	0014
3924	XB,YC,ZC	21	0015
3925	AST STANDBY	15	000F
3926	AST SEARCH	22	0016
3927	AST SEARCH LFOV	23	0017
3928	AST RESET DEFECTS	24	0018
3929	AST LED ON	25	0019
3930	AST LED OFF	26	001A
3931	AST LIGHT FLOOD ON	27	001B
3932	AST LIGHT FLOOD OFF	28	001C
3933	AST FRAME START	29	001D
3934	SET GMT	30	001E
3902	SELF TEST	31	001F
TBD	DRIRU CHANNEL A HIGH	32	0020
TBD	A LOW	33	0021
TBD	B HIGH	34	0022
TBD	B LOW	35	0023
TBD	C HIGH	36	0024
TBD	C LOW	37	0025
3903	STANDBY	1	0001
3904	OPERATE	2	0002
3905	DRIRU ONLY	3	0003
3909	COMET TRACK	7	0007
3911	CALIBRATE	9	0009
TBD	Mirror Reset	48	0030

TABLE B-11: GENERALIZED COMMAND (DATA)

:CMD: WRI-sid-F00x-dddd :ENTER:

SID	COMMAND	WRI
TBD	SET AST DEFECTS	F000 F002 dddd
TBD	AST TEST COMMAND	F000 F003 dddd dddd
TBD	DUMP AST	F000 F004 dddd
TBD	DUMP DEP	F000 F005 dddd dddd
TBD	DUMP PCC	F000 F006 dddd
TBD	AST UPDATE INTERVAL	F000 F007 dddd

TABLE B-12: RAU SYNCHRONOUS SERIAL

SID	COMMAND	WRI
TBD	GMT	F001 dddd dddd dddd dddd
TBD	COMET TRACK	F000 F008 dddd dddd dddd dddd dddd dddd dddd

TABLE B-13: SEID DISCRETE OUTPUTS

SEID -DO	FUNCTION
0	Master Clock Status
32	AST SA Heater On
33	Off
34	Temp CAL Input
48	DRIRU A Power On
49	A Off
50	B On
51	B Off
52	C On
53	C Off
54	DRIRU Heater Power On
55	Off
56	IMCE Power On
57	Off
58	IMCE Heater On
59	Off
60	AST Power On
61	Off
62	AST EA Power On
63	Off

TABLE B-14: PDSS/SEID GML

CYCLE	COMMAND	COMMENT
1	WRITE 1,GMT,1	Broadcast GMT
2	WRITE 0,GMT,1	Broadcast GMT
3	READ 0	Read PCM Channel 0
4	TIME	Read GMT & MET
6	SSEN-BLK 0,1,2,3,4,5,6,7	Read SPSME DI's
8	SSAM-BLK 0,1	Read SPSME AI's
10	SSREAD	Read SPSME Serial
50	PSAMPLE 0	Read RAU FI's
	PSAMPLE 2	
	PSAMPLE 4	
	PSAMPLE 6	
	PSAMPLE 8	
	PSAMPLE 10	
	PSAMPLE 12	
	PSAMPLE 14	
60	PSAMPLE 16	Read RAU FI's
	PSAMPLE 18	
	PSAMPLE 20	
	PSAMPLE 22	
	PSAMPLE 24	
	PSAMPLE 26	
	PSAMPLE 28	
	PSAMPLE 30	
70	PSAMPLE 32	Read RAU FI's
	PSAMPLE 34	
	PSAMPLE 36	
	PSAMPLE 38	
	PSAMPLE 40	
	PSAMPLE 42	
	PSAMPLE 44	
	PSAMPLE 46	
80	PSAMPLE 48	Read RAU FI's
	PSAMPLE 50	
	PSAMPLE 52	
	PSAMPLE 54	
	PSAMPLE 56	
	PSAMPLE 58	
	PSAMPLE 60	
	PSAMPLE 62	

The SEID GML is stored on the PDSS disk under filename 'RFC.MON'.

TABLE B-15: COMET TRACK SEQUENCE DEFINITION

STATEMENT #	STATEMENT
1	IF D[0]<>0 THEN
2	LOOP D[0]
3	WAIT 0,10
4	END LOOP
5	DWRITE 0,9,1
6	ELSE
7	WAIT 10,0
8	ENDIF
9	START 5

NOTES:

1. The Comet Track sequence is stored on the PDSS disk under filename 'RFC.S5'.

2. The Comet Track seqeunce is loaded by PDSS and executed as sequence 5 in SEID ('DEF 5').

3. The Comet Track sequence executes continuously once started. Based on the value of SEID dynamic table entry 0 (D[0]), the sequence performs as follows:

D[0]	SEQUENCE
0	No I/O, Runs every 10 seconds
1	Writes Comet Track data every 10 milliseconds
10	Writes Comet Track data every 1 second

TABLE B-16: NEA LOOKUP TABLE
(CONTINUED)

NOTES:

Given a star Brightness B(x), the Noise Equivalent Angle (NEA) and variance are computed from a table lookup.

Pixel Scale Factor

P2R = 24.51 Arcsec/pixel = 1.1882783E-4 Radians/pixel
(4.8481368E-5 Radians/arc-sec)

Boresight Coordinates

BSC = 239.6 (Column)
BSL = 290.1 (Line)

AST Validity Check Parameters

TOLB = 2 (Brightness Units)
TOLM = 2 (Motion Pixels)

TABLE B-16: NEA LOOKUP TABLE

BRIGHTNESS	NEA	VARIANCE	VARIANCE
1	0.68	0.4624	1.0868E-11
2	0.46	0.2116	4.9735E-12
3	0.29	0.0841	1.9767E-12
4	0.22	0.0484	1.1376E-12
5	0.17	0.0289	6.7928E-13
6	0.14	0.0196	4.6069E-13
7	0.12	0.0144	3.3846E-13
8	0.11	0.0121	2.8440E-13
9	0.10	0.0100	2.3504E-13
10	0.10	0.0100	2.3504E-13
11	0.09	0.0081	1.9039E-13
12	0.09	0.0081	1.9039E-13
13	0.08	0.0064	1.5043E-13
14	0.08	0.0064	1.5043E-13
15	0.07	0.0049	1.1517E-13
16	0.07	0.0049	1.1517E-13
17	0.07	0.0049	1.1517E-13
18	0.06	0.0036	8.4616E-14
19	0.06	0.0036	8.4616E-14
20	0.06	0.0036	8.4616E-14
21	0.06	0.0036	8.4616E-14
22	0.06	0.0036	8.4616E-14
23	0.05	0.0025	5.8761E-14
24	0.05	0.0025	5.8761E-14
25	0.05	0.0025	5.8761E-14
26	0.05	0.0025	5.8761E-14
27	0.05	0.0025	5.8761E-14
28	0.05	0.0025	5.8761E-14
29	0.05	0.0025	5.8761E-14
30	0.05	0.0025	5.8761E-14
31	0.05	0.0025	5.8761E-14
	(ARCSEC)	(ARCSEC**2)	(RADIAN**2)

TABLE B-17: ECIO VOLTAGE CONVERSION

HEX	DEC	VOLTS	HEX	DEC	VOLTS
7F	127	9.96	80	-128	-10.04
73	115	9.02	8D	-115	-9.02
6C	108	8.47	94	-108	-8.47
66	102	8.00	9A	-102	-8.00
60	96	7.52	A0	-96	-7.52
59	89	6.98	A7	-89	-6.98
53	83	6.51	AD	-83	-6.51
4D	77	6.04	B3	-77	-6.04
46	70	5.49	BA	-70	-5.49
40	64	5.02	C0	-64	-5.02
39	57	4.47	C7	-57	-4.47
33	51	4.00	CD	-51	-4.00
2D	45	3.53	D3	-45	-3.53
26	38	2.98	DA	-38	-2.98
20	32	2.51	E0	-32	-2.51
1A	26	2.04	E6	-26	-2.04
13	19	1.45	ED	-19	-1.45
0D	13	1.02	F3	-13	-1.02
06	6	0.47	FA	-6	-0.47
00	0	0.00			

ECIO: VOLTAGE RANGE = -10.0 TO +10.0
COUNT RANGE = -128 TO +127
CONVERSION FACTOR = .07843137

TABLE B-18: HRM VOLTAGE CONVERSION

HEX	DEC	VOLTS	HEX	DEC	VOLTS
1FF	511	9.99	200	-512	-10.00
1E6	486	9.50	21A	-486	-9.50
1CC	460	8.99	234	-460	-8.99
1B3	435	8.50	24D	-435	-8.50
180	384	7.51	280	-384	-7.51
166	358	7.00	29A	-358	-7.00
14C	332	6.49	2B4	-332	-6.49
133	307	6.00	2CD	-307	-6.00
119	281	5.49	2E7	-281	-5.49
100	256	5.00	300	-256	-5.00
0E6	230	4.50	31A	-230	-4.50
0CD	205	4.01	333	-205	-4.01
0B3	179	3.50	34D	-179	-3.50
099	153	2.99	367	-153	-2.99
080	128	2.50	380	-128	-2.50
066	102	1.99	39A	-102	-1.99
04D	77	1.51	3B3	-77	-1.51
033	51	1.00	3CD	-51	-1.00
01A	26	0.51	3E6	-26	-0.51
000	0	0.00			

HRM: VOLTAGE RANGE = -10.0 TO +10.0
COUNT RANGE = -512 TO +511
CONVERSION FACTOR = .01955034

APPENDIX C

PDSS/IMC

RFC USERS GUIDE

RFC USERS GUIDE

C.1 INTRODUCTION

The PDSS/IMC Reflight Certification software executes as an application of PDSS. The user should reference the following documents for details on the operation of the PDSS/SEID.

PDSS User's Manual
IR-AL-001
Revision 2.1
Intermetrics, Inc.
15 July 1984

SEID II Specifications
IR-AL-007
Revision 1.0
Intermetrics, Inc.
15 July 1984

The user should also be familiar with the DEC RT-11 operating system and the DEC LSI 11/23 processor.

The operation of the Reflight Certification (RFC) software package is described below.

C.2 PDSS POWER UP

The following steps should be followed to power up PDSS.

STEP	ACTION
1	Turn Conrac VDU Power Switch On
2	Turn DSD-880 Power Switch On
3	Turn VT-125 Power Switch On
4	Turn Quantex Line Printer Switch On
5	Turn PDSS Crate Power Switch On
6	Turn SEID Power Switch On

The LSI 11/23 will boot RT-11 from the DSD winchester disk. Standard RT-11 operating system commands can be used including setting date and time.

DATE dd-mm-yy
TIME hh:mm:ss

The RT-11 initialization file "SY:STARTX.COM" sets the date. The DATE command in this file can be changed using standard DEC editor functions.

C.3 PDSS Power Down

The following steps should be followed to power down PDSS.

STEP	ACTION
1	Turn Conrac VDC Power Switch Off
2	Turn DSD-880 Power Switch Off
3	Turn VT-125 Power Switch Off
4	Turn PDSS Crate Power Switch Off
5	Turn SEID Power Switch Off
6	Turn Quantex Line Printer Switch Off

C.4 PDSS/IMC REFLIGHT CERTIFICATION CABLES

The following cables should be connected for Reflight Certification.

SEID	IMCE
J10	RAUI (J1, J3, J4, J5, J6)
J11	TMI (J3, J4)
J9	HRMI (J1, J2)
J1, J7	POWER (J2)
J1, J3	IMCE DIOI (J1)

C.5 RUNNING REFLIGHT CERTIFICATION

The following section covers the commands to start and stop the Reflight Certification application.

C.5.1 RFC Start

The Reflight Certification application is initiated by following operations where "..." denotes keyboard entries.

"@RRFC"	RT-11 Program Load
SEID reset	Reset SEID (see below)
"4"	Selection Option 4
" "	Power-On IMCE (see below)
"INIT"	Start RFC Application

The "@RRFC" operation causes the RT-11 operating system to perform command file "RRFC.COM" to load the PDSS/IMC application program and initiate program execution.

When loaded and started, the PDSS program displays the PDSS Master Display page (Figure C-1) on the VT-125 and opens communication with the SEID on the parallel port. When the PDSS LSI 11/23 has established communication with the SEID, the PDSS Master Display will prompt the user to select the program option.

If the PDSS LSI 11/23 cannot establish communication with the SEID, the operator will be prompted to reset the SEID. The SEID reset prompt is noted as the "RESET SEID" message on the PDSS Master Display and the ringing of the VT-125 bell.

The operator should depress the SEID reset button on the SEID front panel **once.** The PDSS Display page should then return to the "SELECT OPTION" message.

CAUTION: Depressing the SEID reset button when not requested or while program is being loaded causes the program to crash (see section C.5.4).

PDSS option "4" should be selected. The display page is cleared and the prompt "?" is displayed.

If IMCE is powered via the simulated CPD and the PDSS/SEID is cabled to allow SEID control of IMCE power supply, the following commands are used to power the IMCE on and off.

"PULSE 32,ON" IMCE Power On
"PULSE 33,ON" IMCE Power Off

If the IMCE is powered via the CPD, the following SEID commands may be used to power up/down the IMCE or the noted Item Entry may be used.

POWER UP

SEID	ITEM ENTRY	FUNCTION
"PULSE 48,ON"	5	DRIRU A POWER ON
"PULSE 50,ON"	5	DRIRU B POWER ON
"PULSE 52,ON"	5	DRIRU C POWER ON
"PULSE 56,ON"	2	IMCE POWER ON
"PULSE 58,ON"	1	IMCE HEATER ON
"PULSE 60,ON"	6	AST POWER ON
"PULSE 62,ON"	1	AST EA HEATER ON
"PULSE 32,ON"	1	AST SA HEATER ON

POWER DOWN

SEID	ITEM ENTRY	FUNCTION
"PULSE 49,ON"	8	DRIRU A POWER DOWN
"PULSE 51,ON"	8	DRIRU B POWER DOWN
"PULSE 53,ON"	8	DRIRU C POWER DOWN
"PULSE 57,ON"	9	IMCE POWER OFF
"PULSE 59,ON"	10	IMCE HEATER OFF
"PULSE 61,ON"	7	AST POWER OFF
"PULSE 63,ON"	10	AST EA HEATER OFF
"PULSE 33,ON"	10	AST SA HEATER OFF

The Reflight Certification task is initiated by the "INIT" command. This command initiates the automatic initialization of the SEID. The following SEID commands are performed and displayed.

```
->SEID BEING LOADED
TVS
SLOAD RFC.S5
DEF 5
GML-RES 3
MLOAD RFC.MON
XSEND
MON
D[1]=.F008
D[1]=.F000
D[0]=.0001
START 5
PDSS/IMC REFLIGHT CERTIFICATION
```

During this period, the operator should not attempt any keyboard commands.

C.5.2 RFC STOP

To stop the RFC task, the following commands should be entered:

```
=STOP
QUIT
```

C.5.3 RFC QUICK START/STOP

To perform a quick stop of RFC:

```
STOP 5
MOFF
PULSE 33,ON or PULSE 54,ON
```

To perform a quick start of RFC:

PULSE 32,ON or PULSE 56,ON
MON
START 5

C.5.4 RFC FAILURES

During the power on/off sequence, if any of the following conditions arise, a recovery procedure should be used.

1. SEID will not initialize
2. Garbage characters appear on CRT
3. Program does not complete initialization

FAST RECOVERY PROCEDURE:

1. Reset CRT (Depress SET-UP,0)
2. Depress SEID Reset
3. Depress LSI 11/23 BOOT

HARD RECOVERY PROCEDURE:

1. Power Off SEID
2. Power Off PDSS CRATE
3. Reset CRT (Depress SET-UP,0)
4. Power On PDSS Crate
5. Power On SEID

C.6 PDSS/IMC RFC COMMANDS

PDSS/IMC RFC commands are grouped into two categories: RFC DDU simulated commands and RFC system commands. Table C-1 lists the commands for each category.

The general syntax for PDSS/IMC commands is as follows:

=cccc</k> <p1,p2,...pn>

All PDSS/IMC commands must have an equal "=" character as the first character. The "=" character is used by the PDSS keyboard monitor for detecting those commands to be handled by user tasks. Failure to have an "=" as the first character results in a PDSS message, "PDSS-68: INVALID COMMAND".

Embedded blanks are not allowed in the 'cccc'.

The < > brackets denote optional data for commands.

Keys (/k) are optional and may be included with commands.

Parameter data is entered as p1,p2,...pn. Unless otherwise specified, the data is entered in hexadecimal. Leading zeroes are not required. Spaces are allowed between parameters but not within the data itself. Either commas or spaces may be used as separators. The number of parameters is a function of the command.

C.6.1 DDU SIMULATED COMMANDS

The DDU commands provide a simulated DDU keyboard function.

TABLE C-1: KEYBOARD COMMANDS

DDU CATEGORY

Command	Parameters	Function
=I	item-number data ...	DDU Item Entry
=P	pfk-number	DDU PFK Entry
=T	type-character-string	DDU Type Entry
=C	XXX sid data ...	DDU CMD Entry

SYSTEM COMMAND CATEGORY

Command	Parameters	Function
=PGMT	day,hour,minute,millisecond	Set GMT
=TASK	task-code	Select Tasks
=CTRL	control-key [,data]	System Control
=VIEW	address	View Memory Data
=TMC		Run Timed Measurement Commands
=LOG	[address,number-words]	Run Log
=STOP		Stop Task
=DISP	page-id	Select Display Page
=PMEM		Print Display Page
=SRST		System Reset
=STAR		Start
=COMM	comment-character-string	Enter Log Comment
=MOD	address data [data ...]	Modify Memory
=TASK	task-mask	Task activate

C.6.1.1 <u>I-Item Entry</u>

Syntax: =I item-number data ...

The =I simulates the DDU Item Entry keyboard function.

Item Entries identified for IMCS are defined in Table B-9 and Figure A-6.

Table C-2 summarizes the Item Entry commands for IMCS.

The IMCS flight display page can be viewed on the PDSS VDU display page 1 (=DISP1).

C.6.1.2 <u>P-PFK</u>

Syntax: =P pkf-number

The =P simulates the DDU PFK keyboard function. No PFK commands are identified for IMCS. The =P is null processed.

C.6.1.3 <u>T-TYPE</u>

Syntax: =T data

The =T simulates the DDU TYPE keyboard function. No TYPE commands are identified for IMCS. The =T is null processed.

TABLE C-2: ITEM ENTRY SUMMARY

ITEM	PARAMETERS	FUNCTION		
1		HTRS ENA		
2		IMCE PWR ON		
3		IMCE LOAD		
4		SELF TEST		
5		DRIRU PWR ON		
6		AST PWR ON		
7		AST PWR OFF		
8		DRIRU PWR OFF		
9		IMCE PWR OFF		
10		HTRS INHIBIT		
11		IMCE STBY		
12		IMCE OPEN		
13		IMCE DRIRU		
14		IMCE CMTRK		
15		CAL		
16		AST DUMP		
17		DEP DUMP		
18		PCC DUMP		
19	aaaa bbbb	START		
			aaaa	bbbb
		AST	0000	AST address(hex)
		DEP	blank(hex)	offset(hex)
		PCC	0000	PCC address(hex)
20	cccc	LNGH=length in words (decimal)		
21		EXEC		
22		MIRROR RST		

C.6.1.4 C-CMD

Syntax: =C WRI sid data (Table B-11)

=C ISS sid (Table B-10)

The =C simulates the DDU CMD keyboard function.

CMD sid's are identified in Tables B-10 and B-11. These commands are distinguished by commands that pulse discretes ("ISS") and commands that write serial commands to the AST ("WRI").

Example:

To select GYRO channels XA,YB,ZA the operator enters:
=C ISS 3917 <CR>

Example:

To add defect coordinates C=10, L=14 the operator enters:
=C WRI tbd-sid F002 0A0E <CR>

Example:

To send an AST test command, the operator enters:
=C WRI tbd-sid F003 dddd dddd <CR>

The test commands are summarized in "Software Requirements Definition for ASTROS Star Tracker (AST) Firmware (DM05, Rev. C), 1 June 1984, Jet Propulsion Laboratory, Figure 2b.

C.6.2 SYSTEM COMMANDS

The System commands identified in Table C-1 provide operator control of system functions. Each command is described in the following sections.

C.6.2.1 COMM Command

Syntax: **=COMM commstr**
commstr = character string of length 16

The COMM command allows the operator to enter a 16 character comment line in the log buffer. On each log cycle, the entire log buffer including the comment field is written to disk.

The COMM command can be used for reference points, reminders, or test headers.

C.6.2.2 CTRL Command

Syntax: **=CRTL</k/l.../m>**
k,l,m=[V;M;E;T]

The CTRL command provides system level control to the operator.

TABLE C-3: TIME VARIABLES

VARIABLE	DEFAULT(SECS)	FUNCTION
T1	1.0	
T2	2.0	
T3	1.0	
T4	1.0	
T5	10.0	
T6	2.0	TEST-MMU LOAD
T7	5.0	AUTOMATIC COMMAND TIMEOUT
T8	1.0	
T9	1.0	
T10	1.0	
T11	1.0	
T12	1.0	
T13	1.0	
T14	1.0	
T15	1.0	
T16	1.0	
T17	1.0	
T18	1.0	
T19	1.0	
T20	1.0	TA
T21	1.0	TASK 21 - COMET TRACK
T22	1.0	TASK 22 - CREW FUNCTIONS
T23	1.0	TASK 23 - DDU PAGE UPDATE
T24	1.0	TASK 24 - EXCEPTION MONITORING
T25	1.0	TASK 25 - ECAS FUNCTIONS
T26	1.0	TASK 26 - LOG FUNCTION
T27	1.0	TASK 27 - DISPLAY UPDATE

PRECEDING PAGE BLANK NOT FILMED

PAGE 120 INTENTIONALLY BLANK

C.6.2.3 DISP Command

Syntax: =DISP</l>pid
l=[I;F;U]

The DISP command is used to request the active display of a display page, to re-initialize a display page, to freeze a display page, or to unfreeze a display page.

Unless frozen, all display pages are updated on a round robin basis at the display rate.

The pid parameter designates the display page (i.e., $1 \leq$ pid ≤ 5). A value for pid outside this range is treated as an invalid parameter and the command is not processed.

Example:

=DISP 2

Requests an active display of page 2. The requested page is mapped to the active page of the VDU.

Example:

=DISP/I 3

Re-initializes the background data from disk for page 3. The foreground or variable data for page 3 will be lost.

Example:

=DISP/F 1

Freezes display page 1. The display function will not update the page data until an unfreeze is invoked.

Example:

=DISP/U 1

Unfreezes display page 1.

C.6.2.4 LOG Command

Syntax: =LOG [addr,number-words]

The =LOG command toggles the PDSS/IMC log control switch between active/inactive. When active, the PDSS/IMC log function logs the IMC Data Buffers to disk file (IMC.LOG) at the time interval [T26=1.0 seconds]. When inactive, the PDSS log function is not performed.

If no parameters are specified, the log function defaults to addr (GMT),852. The log record is 852 words in length and starts at the data entry GMT.

C.6.2.5 MOD Command

Syntax: =MOD adr,hexd,...,hexd
adr = octal address
hexd = hexadecimal data

The MOD command is used to change data. The hexadecimal data is moved into the data buffer beginning at the address (adr) specified. If the address range is actively being displayed on the VIEW page, the display data will be updated.

After all data has been deposited in memory, the next deposit address is displayed on the system console.

C.6.2.6 PMEM Command

Syntax: PMEM <pid<,pid,...>>
pid = page id; $0 \leq pid \leq 6$

The PMEM command prints the display pages on the PDSS line printer. This command provides a hard copy mechanism for saving the display pages during testing. All display pages are printed if no specific pages are requested.

Below are the pages that are available:

pid	Page Printed
0	Active Display Page
1-5	Display Pages 1-5
6	SEID Display Page
blank	All Pages

C.6.2.7 STOP Command

Syntax: =STOP

The STOP command closes the log file, stops the logging function, and clears the CAMAC CSR, INT and CCR registers. The STOP command should be used just prior to terminating a session.

C.6.2.8 VIEW Command

Syntax: =VIEW</S> <adr>
adr = octal address

The view command causes the PDSS/IMC Data or the SEID Data Buffers to be displayed to the VDU. Figure A-11 shows the format of the VIEW display page. The data is displayed as 4 hex characters (16 bits).

The /S control key causes the SEID Data Buffer area to be displayed. If the /S control key and the adr parameters are absent, the VIEW defaults to the ABEGIN area.

The VIEW display page is displayed to the VDU when the =VIEW command is entered. The data on the display is refreshed at a 1.0 second display refresh rate.

C.6.2.9 TASK Command

The TASK command allows the operator to engage or disengage the application tasks. The tasks are selected by the task-mask parameter which is described below.

TASK-MASK:

```
 UUU UUUU UUUU UUUU
 SSS SSSS SSSS SSSS
RRRR RRRR RRRR RRRR
1111 1112 2222 2222
3456 7890 1234 5678
---- ---- ---- ----
```

USR	NAME	RATE	FUNCTION
USR20	EXEC	1HZ	Executive
USR21	COMTRK	10HZ	Comet Track
USR22	CREW	1HZ	Process Crew Commands
USR23	FLTDIS	1HZ	Update DDU Page
USR24	EXMON	1HZ	Exception Monitor
USR25	ECAS	1HZ	Perform IMCS ECAS
USR26	TLOGER	1HZ	Log Function
USR27	QTDISP	1HZ	Update IMCE Displays
USR28	QTKB	1HZ	Keyboard Handler

C.7 MESSAGES

The following messages are displayed to the PDSS system console. An explanation of each message is given.

MST#	MESSAGE
1	INVALID PARAMETERS The command syntax is incorrect, a parameter value is invalid, or the number of parameters is incorrect.

2 INVALID COMMAND
The command is invalid and is not processed.

3 ERROR MAPPING EXTENDED MEM
The RT-11 system calls to establish Extended Memory mapping indicates an error. This is an RT-11 or hardware error. PDSS/IMC will not run without Extended Memory Mapping.

4 LOOPUP ERROR
A system LOOKUP for a data file was in error.

5 READ ERROR
Disk read error occurred.

6 CANNOT OPEN IMC.LOG
The IMC log file (IMC.LOG) could not be opened.

7 LOG FULL
The IMC log file (IMC.LOG) is full and has been closed.

8 PMEM LP ERROR
An error was encountered in writing to the line printer. Verify that the printer is on.

C.8 PDSS/IMC GENERATION

The PDSS/IMC files are as follows.

FILE	CONTENTS
IMCRFC.MAC	RFC Source Code
IMCRFC.OBJ	RFC Object Code
RFC.MON	RFC SEID Montor File
D.001	RFC Display Page 1 Background
D.002	RFC Display Page 2 Background
D.003	RFC Display Page 3 Background
D.004	RFC Display Page 4 Background
D.005	RFC Display Page 5 Background
IMC.LOG	IMCLOG
RFC.S5	RFC Comet Track Sequence

The RT-11 command to recompile the RFC software is:

```
MACRO IMCRFC
```

The RT-11 command to link the RFC software is:

```
@LRFC
```

The contents of the LRFC.COM file is as follows:

```
R LINK
PDSSRFC, PDSS=PDSS, READKB, USRKB, LOG, INTHEX/C
VRAMC, SEID2, USRDP, USRSQ, USRRFC, IMCRFC//
```

The RT-11 command to run the RFC software is:

@RRFC

The contents of the RGT.COM is as follows:

RUN ICAMAC
FRUN PDSSFG.SAV
RUN PDSSRFC

C.9 LOG DUMP

The LDUMP program displays the log on the PDSS CRT.

The operations enumerated below should be followed:

(1) RENAME IMC.LOG ZSEID.LOG
(2) LDUMP
(3) SET-UP 9
log display
(4) SET-UP 9

The NO-SCROLL key can be used to control the display scroll; i.e., to start and stop the display scrolling.

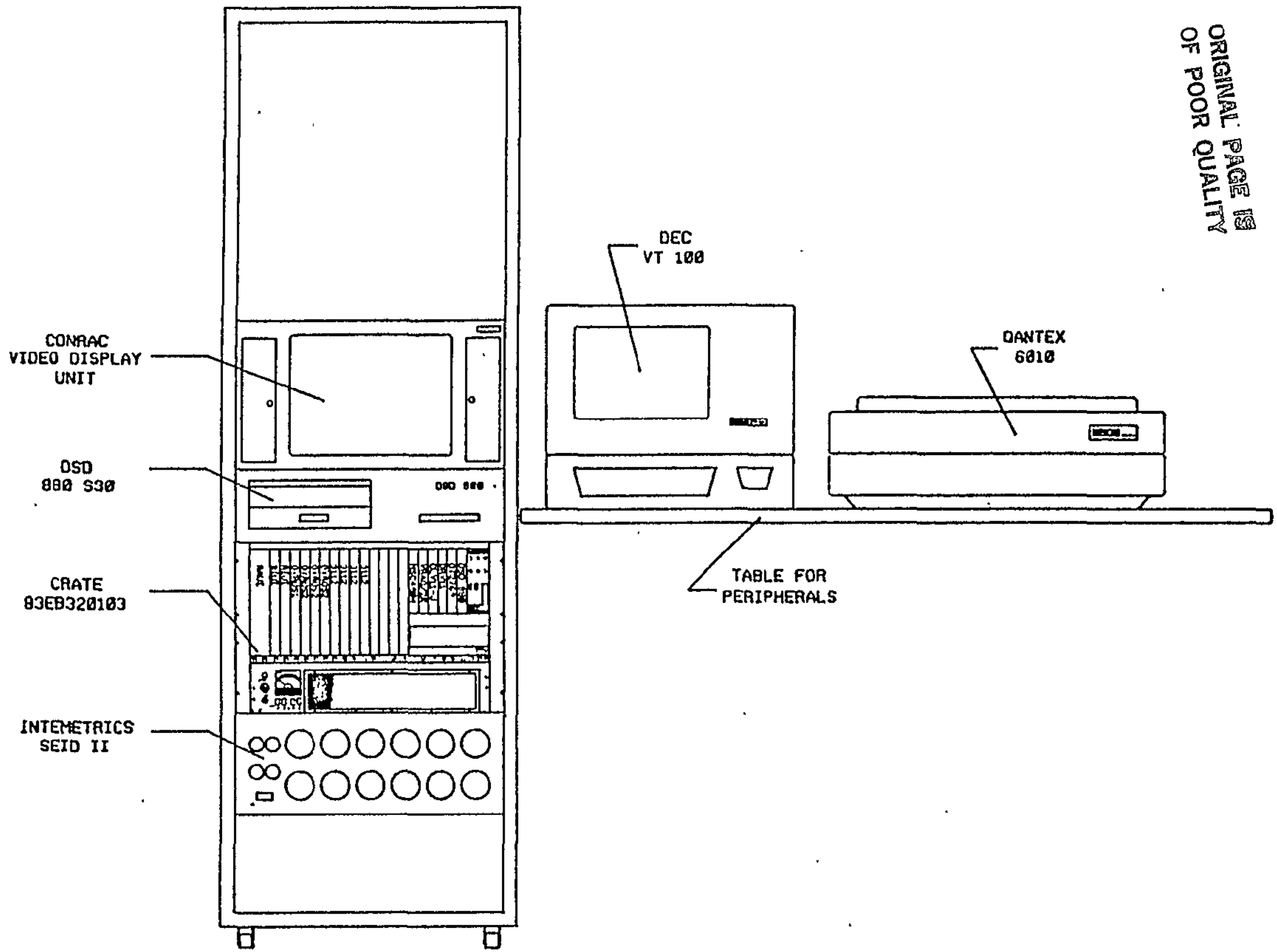


FIGURE C-1: PDSS/IMC GSE LAYOUT

ORIGINAL PAGE IS
OF POOR QUALITY

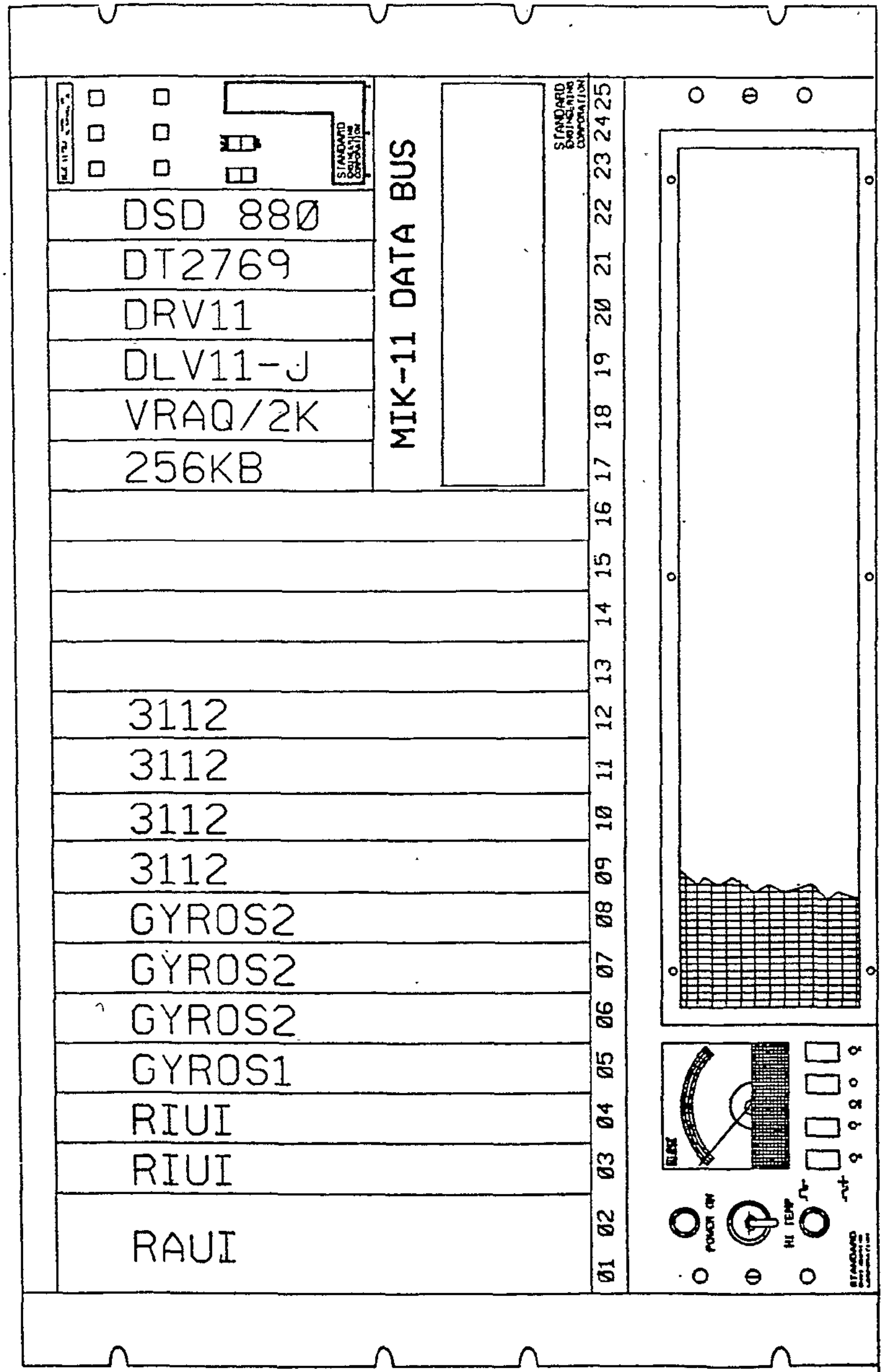
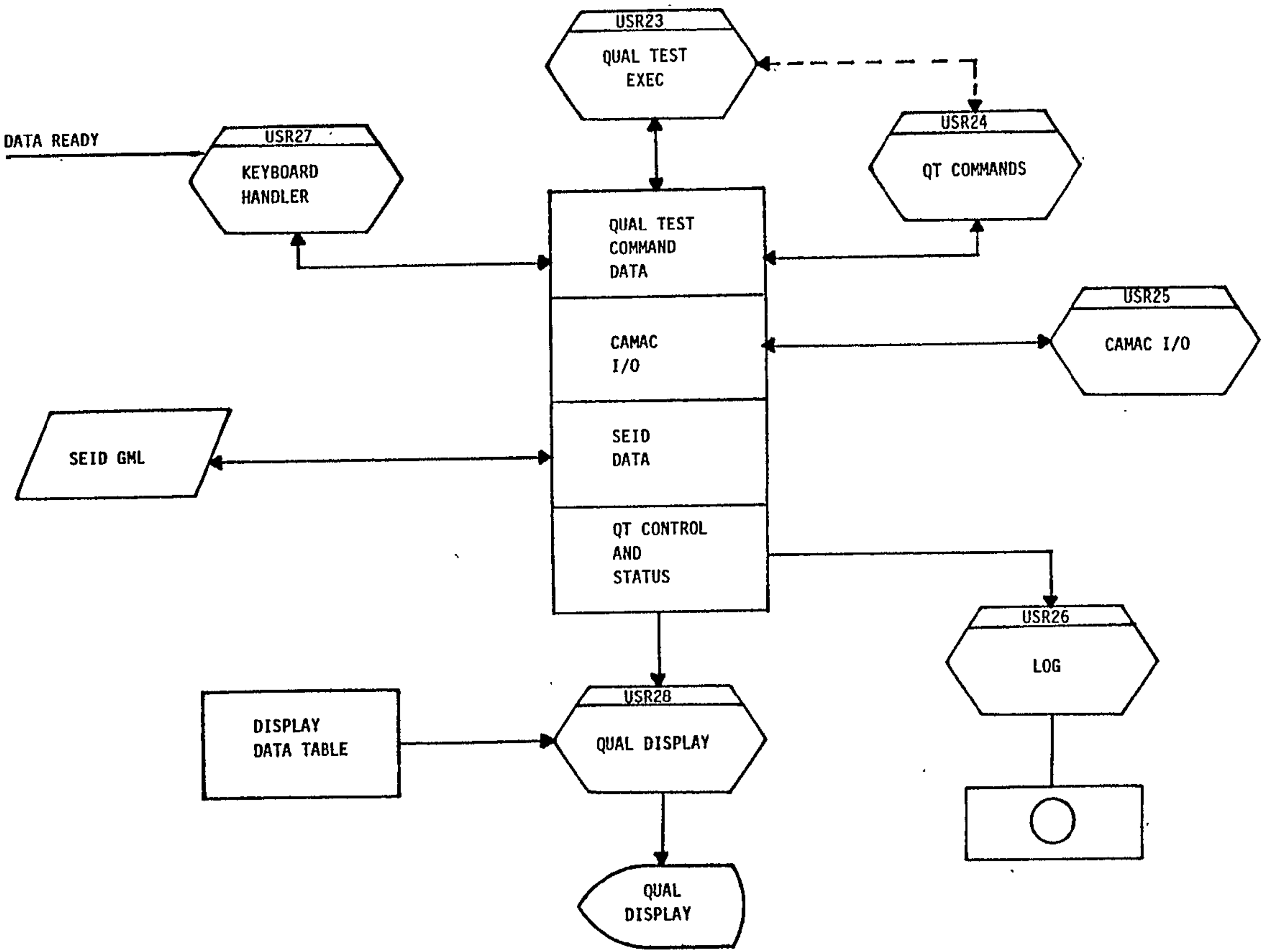


FIGURE C-2: PDSS/IMC CAMAC CRATE



ORIGINAL PAGE IS OF POOR QUALITY

FIGURE C-3: RFC TASK FLOW

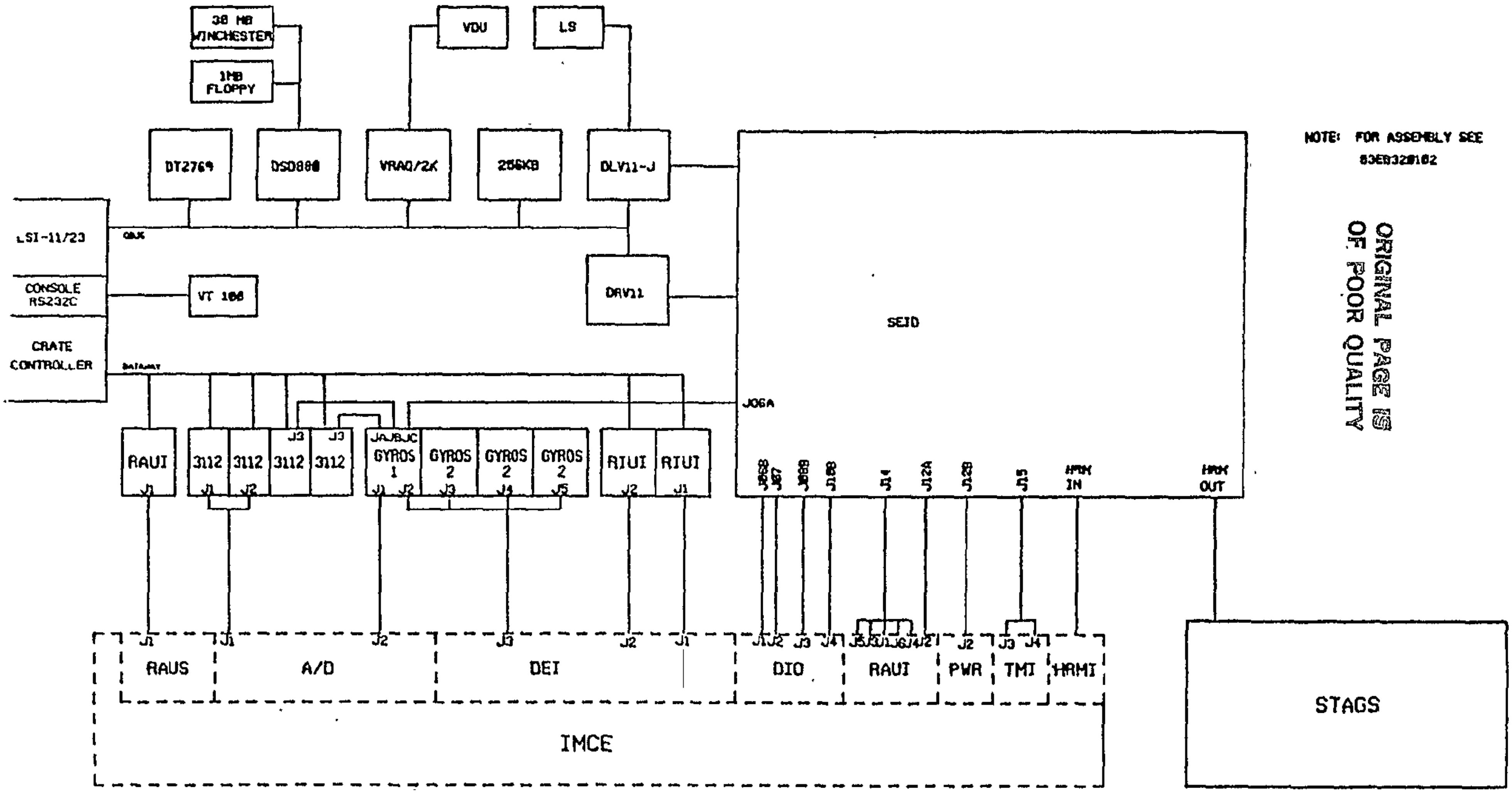


FIGURE C-4: QT INTERFACES

INTERMETRICS